EMMC-CSA
European Materials Modelling Council

BUSINESS MODELS AND SUSTAINABILITY FOR MATERIALS MODELLING SOFTWARE

EMMC White Paper

by Gerhard Goldbeck, Alexandra Simperler (Goldbeck Consulting Ltd)

TABLE OF CONTENT

The EMMC-CSA project has received funding from the European Union's Horizon 2020 research and innovation programme under Grant Agreement No 723867

1. Executive summary

Successful software for materials modelling has an expected lifetime of many decades. This long-term nature requires a sound legal and business foundation: the ownership of software must be clearly established and the license models need to be carefully thought through to ensure a sustainable development and maintenance of the software and impactful exploitation by both academic and industrial end-users. Different business models carefully need to be considered when developing a strategy for long term sustainability of software and sustainment of the operation.

As background, the White Paper provides an overview of materials modelling software market, considering different segmentations (by value chain and by type of modelling) and briefly discusses market dynamics. It is, however, not an extensive market analysis.

Furthermore, the fundamental aspects of software sustainability and sustainment are described, based on literature and previous reports published by the Research Data Alliance and Software Sustainability Institute. In that context, Business Models are discussed, with a detailed analysis of different Revenue Models.

Based on the above background analysis, the status of materials modelling software is presented with respect to different sustainment attributes (Users and Communities, Product Management, Software development and maintenance, Revenue Generation). The findings are based on a workshop and surveys carried out by the EMMC during 2018. Finally, the thoughts and recommendations shared by Software Owners (SWOs) during the evidence gathering are summarised.

The main findings of the White Paper are:

- A variety of business models are used by SWOs, mostly based on a hybrid software and services approach. The revenue percentage share of services varies greatly; it is typically higher in the initial development phase of a software to enable industrial take-up.
- Software sales as well as subscription licenses in combination with a range of services (from initial implementation to contract research) are the predominant revenue mix.
- Services play a significant role, with income ranging from 20-80% in many cases. Target software to services ratio is in the range of 70-80 / 30-20. Services are not as scalable but a substantial amount seems required due to the complexity of the software and science. However, there are also exceptions, with some SWO running a successful business with a pure software (and some training etc.) focus.
- SaaS is still in its infancy. Ways of overcoming industry reservations with SaaS (e.g. security concerns) should be found since SaaS can greatly reduce software maintenance costs and provide a faster route for new features to get to users. Also, SaaS would help to reach small and medium enterprises.
- There is some skepticism but also opportunity for Marketplaces. The added value of the marketplace needs to be demonstrated to SWOs as well, in particular regarding the relation to customers. A concern is that the relation could become more distant rather than closer.

- New businesses developing services or SaaS based on proprietary software is somewhat hindered by the lack of business and licensing models between SWO and SaaS provider as SWO tend to focus on licensing to end users directly.
- Sustainability of software requires a change in education and better recognition of the persons in charge.
- Lifecycle of software requires substantial rethinking and a vision for the future as software's age reaches decades.
- Working with customers (via services and consortia etc.) is important to uncover why they are using your software and what it takes to retain them as well as to fund new developments.
- Government funded projects are also important to most SWOs for development.
- Most software represented is proprietary but there are a range of involvements with open-source. There is a lot of complementarity (e.g. pre- post- processing and materials relations for open source codes) and SWOs can profit from the collaborative opportunities it brings.
- It is important to engage with the academic community, find ways to make software engineering more exciting and bring in new standards to make software sustainable and maintainable.

2. Evidence basis

2.1 Software Owners Expert Meeting

A workshop with a focus on commercial Software Owners (SWOs) was held on the 4th of July 2018 in Cambridge. About 25 SWOs covering a variety of organisation sizes and business types were invited, ensuring that all types of models were covered. The meeting was attended by 14 delegates including 11 different SWOs covering a wide range of businesses from start-ups to established ventures and all types of models: electronic, atomistic, mesoscopic and continuum models were each covered by between four and six participant organisations. Seven attendees were from outside of the EMMC-CSA, and seven from EMMC-CSA partners.

Successful software for materials modelling has an expected lifetime of many decades. This long-term nature requires a sound legal and business foundation: the ownership of software must be clearly established and the license models need to be carefully thought through to ensure a sustainable development and maintenance of the software and impactful exploitation by both academic and industrial end-users. Solutions need to be considered for different forms of software including individual software components, comprehensive codes and software platforms. Various business models should be analysed ranging from those built on services around open source software to proprietary solutions.

Three main topics were covered by the workshop:

1. Experiences with different business models: to see the status quo
2. Marketplaces: to see how these can aid the software owners
3. Sustainability, maintenance and product life cycle management: to learn about the efforts and find best practice

Webpage of the workshop:**https://emmc.info/events/emmc-csa-expert-meeting-sustainable-economic-framework-for-materials-modelling-software/**

2.2 Software Owner Interviews

Interviews were conducted with eight SWOs from academia and government facilities (referred to as academic SWOs) and 19 SWOs from companies (including non-profit) (referred to as commercial SWOs). The types of models covered are: electronic (11 SWOs), atomistic (9 SWOs), mesoscopic (4 SWOs), continuum (11 SWOs) and data-based (8 SWOs), see Figure 1. The companies ranged from start-ups to very large corporations. As shown in Figure 2, 47% of commercial SWOs work with less than 10 persons on their software, 42% have between 10-100 employees and 11 % employ more persons.

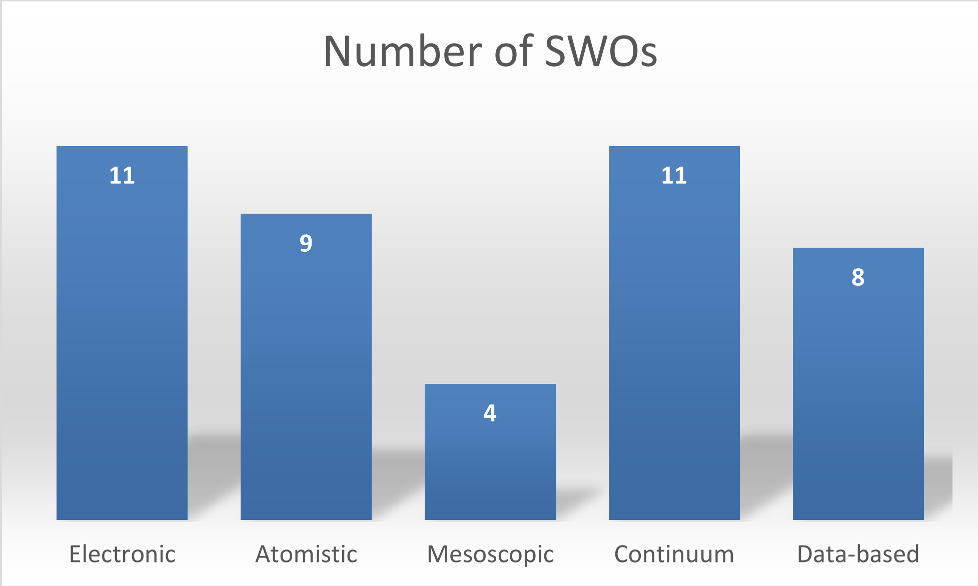

Figure 1: Number of Software Owners interviewed according to the type of model. For model classification, see [CEN Workshop Agreement: Materials modelling – Terminology, classification and metadata] and [Review of Materials Modelling].

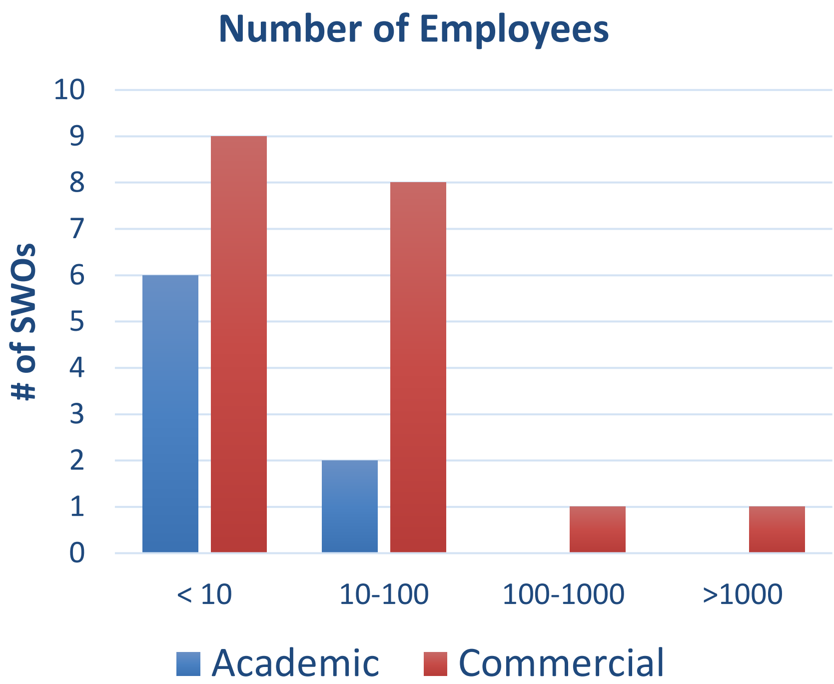

Figure 2: Number of Employees (commercial SWOs) and Developers (academic SWOs)

For comparison, for academic SWOs the number of developers working on their software is shown. Most work with one to 10 persons. Academic developers may be located in the research groups of the SWO but also at other collaborating institutions.

Software revenue/income distribution is shown in Figure 3. For the commercial SWO a revenue of zero means that they are just starting their business. The largest number falls into the bracket of less than one million Euros per year. In the case of academic SWO, no income means that their software it is distributed for free. As in the commercial case, most academic SWO also fall into less than a million Euros bracket.

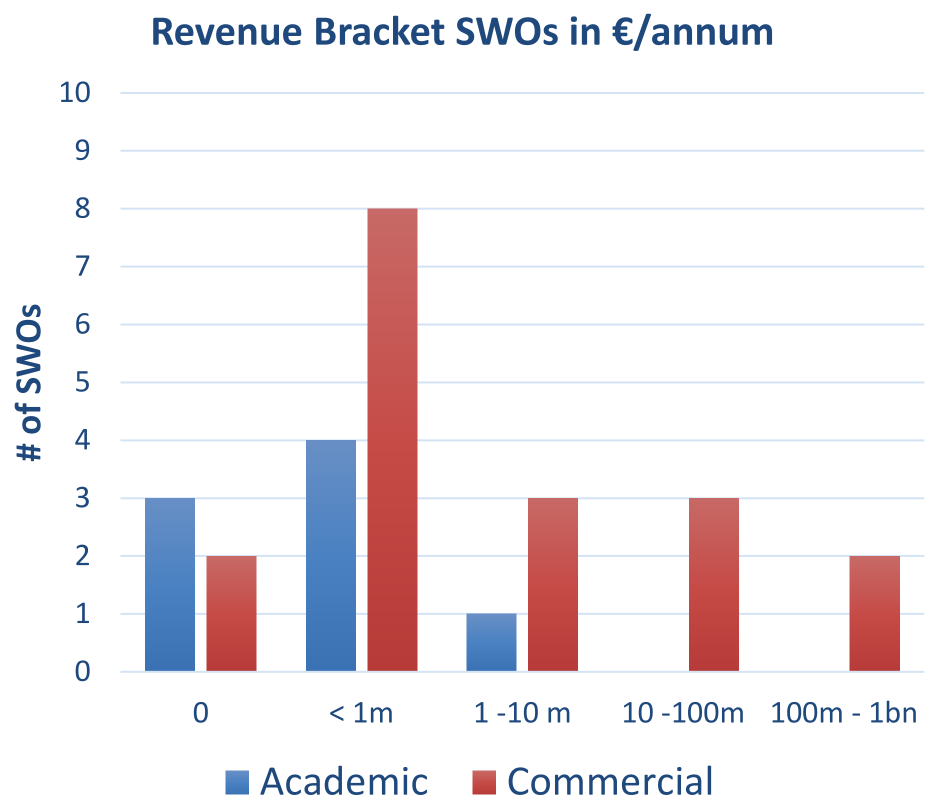

Figure 3. Revenue Bracket in €/annum.

2.3 Software User Survey

It was pertinent to this study to complement the SWO perspective with that of Software Users (SWUs). Surveys from users of materials modelling software from 14 companies representing different industry sectors (referred to as commercial SWUs), as well from 13 universities and research organisations (referred to as academic SWUs) were obtained. The survey included 11 questions to learn about their expectations with respect to materials modelling software (licensing, services, support, costs, etc.) and what they see as the value and as the barriers regarding the uptake of materials modelling.

2.4 Literature review

A review of pertinent literature about software sustainability (see the References section) was carried out and is summarised. In particular, a significant part of a report on sustainable business models published by a Research Data Alliance group [iv] has been adapted.

3. Materials modelling software market

Materials modelling software serves a wide range of markets including Electronics, Consumer Packaged Goods, Chemicals, Materials, Energy, Manufacturing and Life Sciences.
A number of key aspects of the industry and its market is covered in the Goldbeck Consulting report on Scientific Software Industry [i], addressing the following topics:

- The structure of the software industry.
- Requirements for software development: in-house and through collaboration.
- Routes to market for scientific software, e.g. via software houses or direct licensing into specific industries.
- Commercialization requirements: standards, IP ownership, licensing schemes.
- Warranty and liability issues.

There are no market analyst reports specifically focussing on materials modelling. The relevant established market areas that are covered by analysts tend to be Cheminformatics (including in particular discrete modelling) (e.g. https://www.grandviewresearch.com/press-release/global-chemoinformatics-market), which is mainly focussed on drug discovery but includes companies that also sell discrete materials modelling software. The other is Computer-Aided Engineering, which significantly includes continuum materials modelling but also a proportion of computer-aided design (i.e. outside of materials modelling). Examples of reports are:

- https://www.transparencymarketresearch.com/computer-aided-engineering-market.html
- https://www.businesswire.com/news/home/20171020005370/en/Global-CAE-Market-2017-2021-Growth-Application-Specific-CAE
- https://www.zionmarketresearch.com/news/computer-aided-engineering-market

3.1 Market Segmentation

Market segmentation can be done along different dimensions:

1. By considering the position in the value chain where materials modelling is used.
2. By the type of modelling, in particular data-based versus physics-based modelling and for the latter by the entity type described (electronic, atomistic, mesoscopic and continuum).

The two dimensions are actually not completely independent, since at least currently there is more use of continuum models downstream, as Figure 4 illustrates.

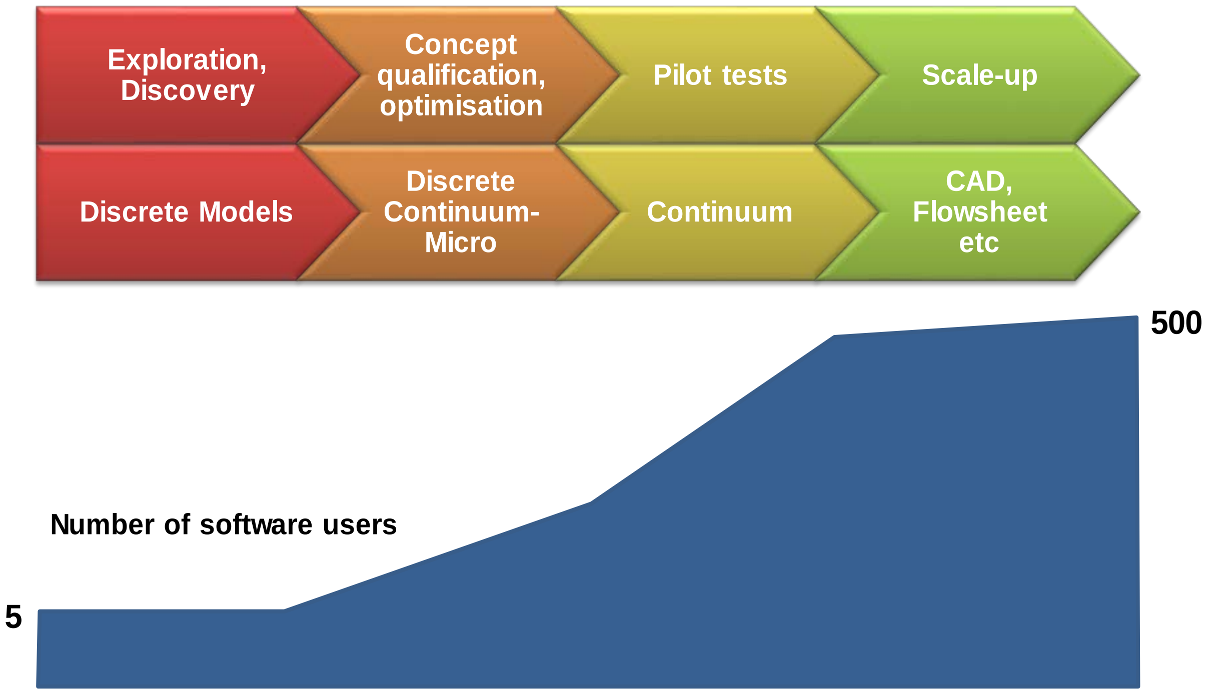

Figure 4: Mapping of Product Value Chain (exploration to scale-up), typical model used (discrete to CAD), and number of software users at a large company.

3.2 Market segmentation by value chain

The development and engineering sector representing the design *WITH* a given material is far larger than the materials research and design sector where the target is to develop *THE* material itself. There are estimates from market research organisations for the size of the Computer-Aided Engineering market, which roughly represents the former (design *WITH* the material) but will also include a lot of design software which is strictly outside of materials modelling. The total market size is about $4bn with a CAGR of about 11%. From public reports of companies which represents "*modelling WITH materials*" we can deduce that the total market for materials modelling software in the downstream sector is more than $1bn.

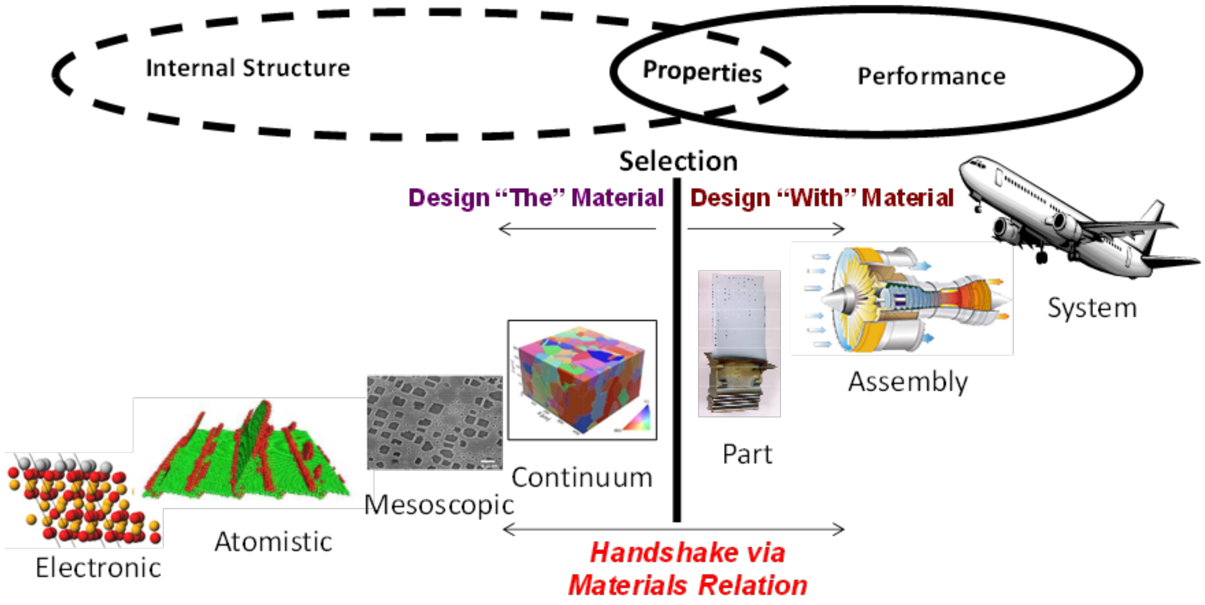

Figure 5: Segmentation of the field into Materials and Engineering Modelling, designing "the" material on the left, and the design of product "with" the material on the right (Courtesy of Granta Design).

A rough estimate of the market size for materials modelling software in the "*design THE material*" sector is in the range of $100m, i.e. about a factor of 10 smaller than the downstream sector. This estimate is based on revenues and numbers of employees in relevant software companies

About half of the Software Owners represented at the workshop focus entirely on providing software and services to industry (e.g. in the case of software that is open and free for academics), while the remainder also sell software to the academic research sector. For the latter, the share of revenue from academia is declining as the commercial sector income grows faster.

One organisation also has a significant market in providing teaching solutions in materials engineering. In terms of industry sectors sold into, all industries including Electronics, Consumer Packaged Goods, Chemicals, Materials, Energy, Manufacturing and Life Sciences are served by one or the other of the participants. While SWOs provide models that are in principle widely applicable, all have certain sectors they focus on (e.g. semiconductor industry, aerospace industry, chemicals industry, etc.)

The Computer-Aided Engineering (CAE) market currently generates global revenues of 1.2 billion €. Also, it is growing at an annual rate of 11% (according to TechNavio's report). The global CAE market is forecast to grow at a CAGR of 11.84% during the period 2017-2021.

Zion Research has published a report titled "Computer Aided Engineering Market (Finite Element Analysis and Computational Fluid Dynamics) for Aerospace, Automobile, Electronic and Electricals, Defence, Industrial Machineries and Other Applications called "Global Industry Perspective, Comprehensive Analysis and Forecast, 2015-2021". According to the report, the global demand for computer aided engineering market was valued at around USD 3,062.37 billion in 2015, and is expected to reach approximately USD 5,863.36 million in 2021, growing at a CAGR of slightly above 11.10% between 2016 and 2021.

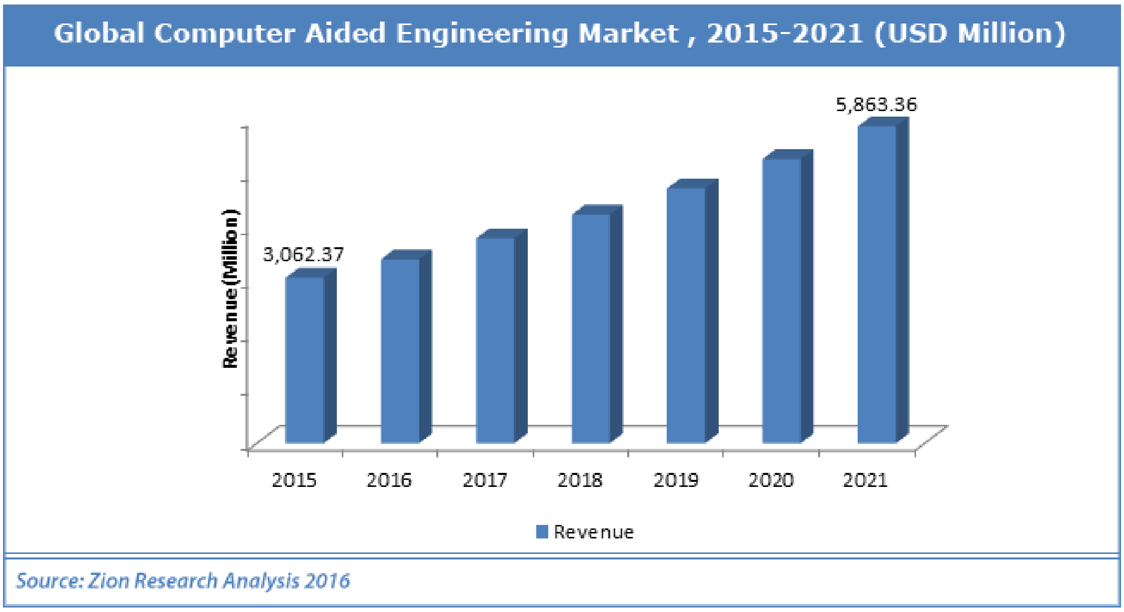

Figure 6. Computer Aided Engineering Market

3.3 Market segmentation by type of modelling

The Discrete Modelling market is estimated to be in the region of € 100m. The share within that market of electronic atomistic and mesoscopic modelling is not documented but anecdotal evidence, backed up by hard evidence such as number of software codes and companies, number of documented users etc. suggests that electronic models make up the largest part, followed by atomistic and then mesoscopic modelling.

The Continuum Materials Modelling Market is a large fraction of the CAE market (which also includes some more design type and data management applications). It is estimated in the region of at least €1bn which makes it an order of magnitude larger than the discrete modelling market.

3.4 Software business dynamics

While a detailed analysis of the dynamics of the materials modelling software market is beyond the scope of this report, some key factors can be observed:

- The continuum materials modelling market is relatively mature in the case of engineering focussed applications (Design with the material), where the focus has been on large scale integration of CAE, CFD etc (i.e. integrating all types of 'physics'), CAD, PLM and even asset management. However, there are many small and relatively fast-growing companies representing the continuum modelling of the material. Some of these companies have also been acquired by the global CAE/CAD/PLM corporations.
- The discrete materials modelling market went through a period of consolidation in the 1990s, followed again by new companies and ventures starting up since then. Some companies have been taken over by large modelling and design corporations but there remain a significant number of smaller players and start-ups.

4. Software sustainability and sustainment

Software sustainability describes the practices, both technical and non-technical, that allow software to continue to operate as expected in the future. A constant level of effort is required to maintain the software's operation [ii].

However, in order to achieve sustainability and maintain a sustainable operation as a Software Owner, a wider range of factors and activities need to be considered. The term sustainment is used to capture them. Sustainment is typically used in the military to mean the provision of personnel, logistic, and other support required to maintain and prolong operations or combat until successful accomplishment or revision of the mission/objectives. In business it has been used in the context of actions required to optimise the value in-use and cost in-use for long-life assets, and develop through-life support capability to meet stakeholder demands.

A working definition of **software sustainment** has been formulated by the Software Engineering Institute (SEI) as follows [iii]:

The processes, procedures, people, material, and information required to support, maintain, and operate the software aspects of a system.

Quoting from Ref [3]: "*Successful software sustainment consists of more than modifying and updating source code. It must address many other issues such as documentation, operations, deployment, community development, security, configuration management, training (users and sustainment personnel), help desk, product management, and technology refresh. It also depends on the experience of the sustainment organization, the skills of the sustainment team, the adaptability of the customer, and the operational domain of the team.*"

Based on the above, a suite of attributes for sustainment has been adapted from the report from the Sustainable Business Models Team to the Brokering Governance Working Group of the Research Data Alliance (RDA) [iv] (see Figure 7). These attributes will take on varying levels of importance during the life-cycle of software although some might not be as important at points in the life-cycle such as the initial development phase, but will receive equal consideration for sustaining mature software. Each of these categories requires some level of financial and human resource investment. When assessing alternative business models for software sustainment, an evaluation can be made as to how well the model fulfils each of these requirements.

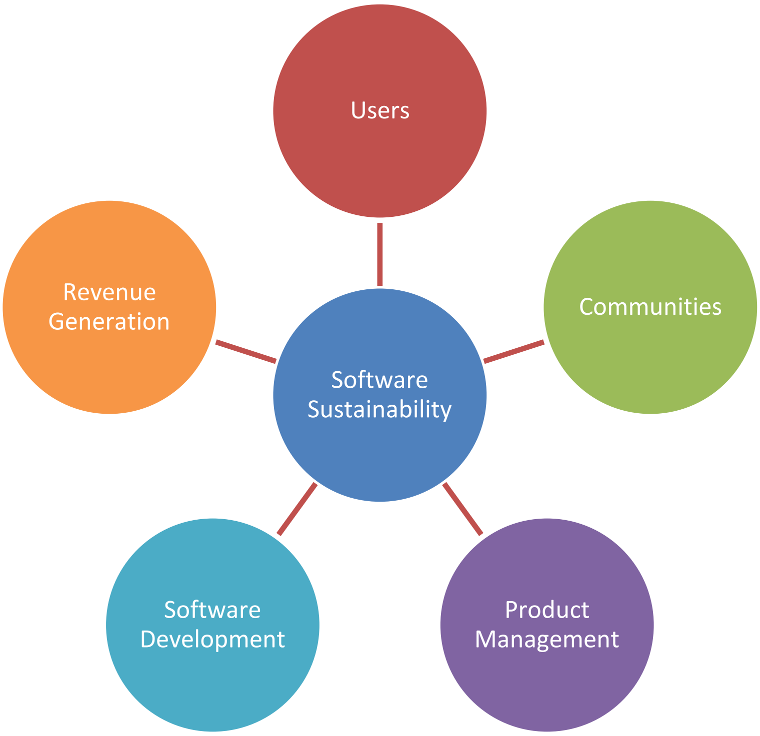

Figure 7: Required attributes for software sustainability adapted from Ref [Fehler! Textmarke nicht definiert.**] and the Software Sustainability Institute's definitions for software sustainability (http://www.software.ac.uk/).**

4.1 Sustainment Attributes

- Users: The efforts required to identify and target, engage and sustain a class of institutional and/or science users.
- Communities: Identification, targeting and engagement of stakeholder groups, including online discussion forums, social media etc. Also includes potential community contributions to the software ranging from code contributions in software source software to testing and validation.
- Product management – The active management and innovation of the product in response to market needs as well as new academic developments: understand requirements, roadmaps, prioritisation of new developments and pursuing the case(s) for commitment and implementation. Also includes licensing, distribution, security etc.
- Software Engineering – The effort, including formal and informal processes, required to provide development and ongoing maintenance, improvement and technology assessment of products.
- Revenue Generation – The efforts required to obtain and sustain funding. This may include software marketing and sales as well as proposal writing, venture capital gathering, reporting etc.

4.2 The Concept of "The Technical Debt"

If you do not make software sustainable from its humble beginnings as a code, you occur "technical debt". The phrase was coined by Ward Cunningham, one of the authors of the Agile Manifesto. *"Shipping first-time code is like going into debt. A little debt speeds development so long as it is paid back promptly with refactoring. The danger occurs when the debt is not repaid. Every minute spent*

on code that is not quite right for the programming task of the moment counts as interest on that debt. Entire engineering organizations can be brought to a stand-still under the debt load of an unfactored implementation, object-oriented or otherwise." Ward Cunningham, 1992 OOPLSA conference.
https://www.agilealliance.org/introduction-to-the-technical-debt-concept/
Technical Debt can naturally occur if a SWO has to push to the market and therefore uses short cuts to create a viable but not necessarily a perfect piece of software that still delivers business value. The issues may be related to coding practices and style, test coverage, missing documentation, potential bugs, etc. The idea is here to monitor where technical debt will occur and then be prepared to gradually pay it back.

5. Business Models

The following compilation and analysis of business models in this report is based on the following

- The seminal essay by Eric Raymond "The Magic Cauldron" [v]
- Report from the Sustainable Business Models Team to the Brokering Governance Working Group of the Research Data Alliance (RDA) [iv]
- RDA discussion paper on Software as a Service business model [vi]
- Publications on marketplaces [vii, viii, ix]

In particular, the detailed discussion of different business models in section 5.3 is based on Ref [iv].

5.1 Overview

5.1.1 Business Model

A Business Model can be defined as the plan for the successful operation of a business, identifying sources of revenue (see 'Revenue Models' below), the intended customer base, products, and details of financing. A Business Model describes the rationale of how an organization creates, delivers, and captures value [x].

5.1.2 Business Roles

Scientific software (including materials modelling software) providers typically serve three business roles:

- A creator that transforms ideas into a product
- A distributor of software
- A lessor that provides the rights to use the software

An underlying premise with respect to software is that the provenance and intellectual property (IP) are established. Software businesses may use a hybrid approach serving more than one role since they act as both the inventor and the IP lessor.

5.1.3 Revenue Models

Software business derive revenue in a number of different ways. A typical business model is based in most cases on a hybrid approach utilising a range of revenue models. Revenue Models relevant to materials modelling software include the following:

- Product Sales and maintenance services
- Subscription based software licensing
- Services and consulting
- Open-source based business models (see section 5.1.4)
- Government funding
- Software as a Service (SaaS)
- Marketplace business models [**Fehler! Textmarke nicht definiert.**] and strategies [**Fehler! Textmarke nicht definiert.**]

5.1.4 Open Source based business models

According to Raymond [v] there are the following business models based on (Free) and Open Source software.

- Loss-leader/market positioner: use open source software to create or maintain a market position for proprietary software (e.g. an open source client creates a market for a proprietary server).
- “Widget frosting”: publish open source drivers for proprietary hardware, both for peer review benefits and also to allow operating system vendors/maintainers to adapt the driver to future changes in system interfaces.
- Consulting: use expertise in an open source product to drive revenue for packaging and/or consulting services (e.g. OpenFOAM).
- Accessorising: sell books or other accessories to open source products (e.g. O’Reilly publishers).
- “Free the future, sell the present”: sell a proprietary product with a license that guarantees open source release after a certain time, in order to guarantee future maintainability to prospective customers (e.g. Alladdin, GhostScript).
- “Free the software, sell the brand”: charge for the branded, trademarked, tested, and certified version of an open source product (e.g. RedHat, SUSE).

Raymond’s essay also pointed out that in most cases, open source software does not have as much revenue available to fund its development as is provided by license fees of proprietary software. For this reason, proprietary software often leads the development of end-user software, while open source provides low-end users with an inexpensive alternative, and also provides an open and flexible platform for those who need custom modifications.

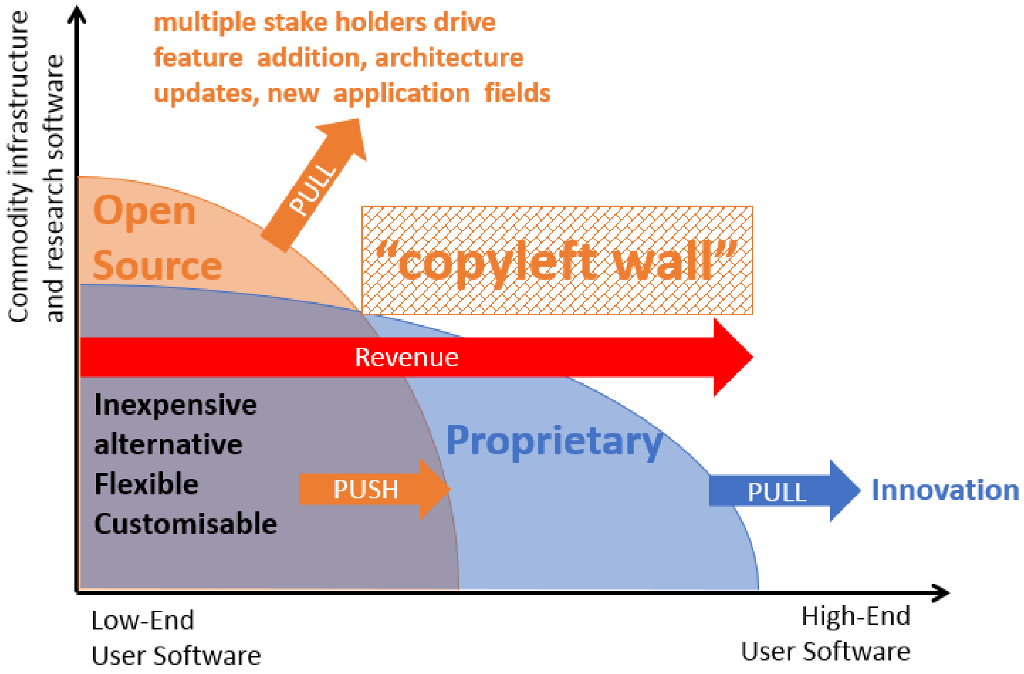

Figure 8: Adaption of Eric Raymond's concept of expansion of open source and proprietary software capabilities.

5.2 Revenue models

5.2.1 Overview

The RDA report [iv] discusses the strengths and weaknesses of different revenue models in the context of long-term sustainability. They find that a business model incorporating aspects of different revenue models over the lifespan will likely provide the strongest model for sustainment. The discussion below of revenue models (based on [iv]) includes the following aspects:

1) Description of the revenue model
2) Mapping Revenue Models onto Sustainment Attributes
3) Characteristics of the revenue model that would benefit the sustainability of software
4) Challenges in the application of the model

5.2.2 Product /Service Sales

Model Description

A product or service revenue model is based on an outright sale of software for customer platforms with the customer having ownership of the software and then potentially implementation and maintenance service to support operations on the customer's platform is provided by the vendor.

Mapping Revenue Models onto Sustainment Attributes

Income: The product/service sales model provides a revenue stream that is directly proportional to the number of items "sold" or delivered.

Users – Market development is a core component of the product/service sales model.
Communities – Engagement with the community of potential users is critical to the success.
Product Marketing – In the product/service sales model there is a requirement for marketing and promotion of the service as visibility and awareness of the service is a prerequisite for its use.
Software Engineering –The system must have capabilities that are well aligned with the needs and requirements of customers.

Advantages

- Capability to provide steady, long-term funding
- Can contribute to a business diversification strategy for resilience

Challenges

- Expectations due to "free/open source" works against buy-in for license costs

5.2.3 Government Funding

Model Description

For decades, governments have provided funding to support the development and sustainment of software. Grants are assistance awards that, in the context of the interests of the Research Data Alliance, are often used to "assist" university and non-profits in the pursuit of a research problem. The supporting government entities generally have little involvement in guiding the research activity that is funded by grants.
This sort of funding, for example, has significantly enabled Australian researchers to participate in eResearch by supporting the development of IT infrastructure, software, tools and services for open publication, discovery, sharing and use of Australia's research data outputs through the Australian National Data Service (ANDS) and National eResearch Collaboration, Tools and Resources (NeCTAR).

Mapping Revenue Models onto Sustainment Attributes

Funding – Requires a proposal. The home institution of the principal investigator bears part or all of the cost of preparing the proposal. Once a grant is awarded, the revenue stream is virtually assured for a limited period of time. Government initiated strategic investments can (in the US in particular) be of longer duration than research grants and cooperative research agreements, but are nonetheless limited in their duration.
Users – The concept of market development may be interpreted to mean understanding which research communities might find the software useful in the pursuit of their objectives. This is an important observation that could have an impact on sustainability.
Communities –Community engagement is often encouraged and sometimes additional funding for outreach and engagement activities may be provided.
Software Engineering – Investing in good software engineering is an important use of the revenue obtained from the Government. However, government funding of a project, particularly in a university setting, does not assure that good software engineering practices will be used throughout the project.
Product Management – Under government funding, product management is usually a stage removed from the concerns of the project. While government funding increasingly encourages exploitation and industrial impact considerations, the developments nevertheless happen at a stage

somewhat removed from industrial users. Intellectual Property rights usually reside with the institution where the software was developed.

Advantages

Advances the conceptualization, prototyping, and initial testing of the software for such attributes as functionality and robustness. There are some instances of longer-term support of software that the funding agencies deem necessary for significant community functions (e.g. weather prediction).

Challenges

- Funding term limits do not create sustainment.
- Government initiated strategic investments may change direction.
- Government support for software development does not ensure that good software engineering practices are employed.
- When software is created for one purpose and used by its developer, ease of operation by others or an outside organization is not generally a concern of the developer.

5.2.4 Software as a Service (SaaS)

Model Description:

Software companies can differentiate their business model by offering software as a product or, software as a service (SaaS), or a combination of both. As K.M. Popp (2015) notes, "*SaaS means the software vendor does not deliver the software, but the customer gets both access to the software and usage rights […] The software vendor carries the cost of software support, maintenance, and operation.*"

A common model for software and services sustainability in the private sector is called "Software as a Service" or SaaS. While there are multiple definitions of SaaS, a common one that is useful for the present discussions is: "software that is owned, delivered and managed by one or more providers and is available remotely". The provider offers access to the software that is consumed in a one-to-many model at any time on a pay-for-use or subscription basis.

- Hosted
- On demand
- Integrated (operates on a platform)
- Subscription or other form of relation
- Multi–tenant (simultaneous use)
- Supports Network effect i.e. builds more rapidly, leveraging marginal benefit and tipping point

The initial service offering can be a free service that engages the community ("Freemium")

Typical Pricing models are:

Pricing Models (includes Freemium)	Description
Capacity-based Model	According to capacity, usage, or number of users reach certain thresholds
Feature-based Model	According to the number of key features available
Time-based Model	Subscription with a fixed (extendable) time

Use-case Model	Fees are based on specified categories of customers (non-commercial use, educational, non-profit, etc)
Quality of Service Model	Customers get a certain level of response time or service level

Mapping Revenue Models onto Sustainment Attributes:

Funding – A complementary mix of contributions mentioned above.

Users –Large institutions and businesses that will benefit from the interoperability offered. Business model must address how to provide enough unique benefits from the service that customers will not simple download the software and apply their own software engineering.

Communities – For customers from the research community, community engagement through presentations at conferences, working sessions, demonstrations and other direct interactions are part of the culture. Community advocacy can support market development.

Community contributions are done on a voluntary basis. For the research community, the mixed model of community contributions and a SaaS modality can be complementary and provide a broad customer engagement for sustainability.

Software Engineering – The role of software engineering is to develop and maintain an operational capability of the software.

Product management – "Traditional" SaaS implementations generally use conventional product management approaches. Documentation for users is usually web based and accessible through the internet. Chat rooms are a common means of offering support.

Advantages:

- cover a wide range of both open source and proprietary software capabilities.
- Flexibility of pricing from very modest costs to more structured implementations.
- Shorter development cycles when software can be pushed out to the cloud with smaller batch sizes, faster feedback and high overall quality.

Challenges:

- New capabilities may not be relevant to a subset of users and they must adapt to the change.
- The research community may be a low volume customer.
- Need for a set of software interfaces or application interfaces to provide a broad interoperability capability.

5.2.5 SWO Sponsored Consortium of Software Users

Model Description

A widely used model in materials modelling software companies is one where the corporate entities takes the lead to establish a Consortium of current and new end users in an area of common interest.

The membership model may provide more direct participation in system governance, depending upon the specifics of the model.

Mapping Revenue Models onto Sustainment Attributes

Funding– By potential corporate sponsors/members; requires a high degree of interaction and maintenance through time.

Users – Market development is a core component.

Communities – Engagement with the community of potential users is critical to the success. Membership provides a way of building a community and an ongoing community engagement.

Software Engineering –must have capabilities that are well aligned with the needs and requirements of the sponsoring organizations.

Product management –Product management model must show some flexibility regarding the needs of members and that of the wider market. There may also be IPR considerations (due to member contributions) and exclusivity periods before software can be released more widely.

Advantages

● Capability to provide steady, long-term funding

● Can contribute to a business diversification strategy for resilience

● Could build partnerships with membership organizations to provide premium level engagements

Challenges

● May require significant promotion and effective communication with corporate sponsors

● Tension between member requirements and product management for wider market

5.2.6 Information and Ad Sales

Information and Ad Sales can be an important contributor to revenue for marketplaces.

Model Description

Online information and ad sales are forms of marketing that use the Internet to deliver marketing messages to consumers. This could potentially be providing an additional revenue stream. This is how Google, Facebook, YouTube operate.

Mapping Revenue Models onto Sustainment Attributes

Funding – Funding is provided by the business or industry that benefits from information and Ads.

Users – Considerable effort may be required to identify businesses or industries that desire the information that can be obtained and to identify appropriate advertisers.

Communities – Caution must be exercised in selling user information.

Software Engineering – Software development or contracts with third parties may be necessary to capture/package information and support advertisements.

Product Management –Requires significant relationship management. Additional resources may be required to capture and package data and information desired by external business and industry interests.

Advantages

● automated mechanism and requires little human intervention

● scales easily as user numbers increase

● may include a user model where users pay to not see ads

● there is potential for annotation service provided by this model to become part of institutional quality assessment process

● this model can contribute to a business diversification strategy for resilience and growth

Challenges

● requires sales to advertisers and/or people that wish to use the service -- requires a sales team and an identified large consumer base
● requires good planning of communications and branding
● ads may disenfranchise user base
● sales of information are a major privacy issue and may disenfranchise user base
● questions about who owns the usage and annotation data

6. Materials Modelling Software Sustainment

In the following, results of the evidence gathering from interviews, surveys and workshop are presented. The results are organised along the sustainment attributes of Users and Communities, Product Management, Software Development, and Revenue Generation, with an additional section on Marketplaces.

6.1 Users and Communities

Engagement of users and communities is regarded as highly important by Software Owners. In fact, as will be seen below, user engagement is also a key driver to adopting relevant revenue generation models, in particular service-based engagements and User Consortia.

Some of the SWOs aim to provide a full/holistic solution for their customers, offering a range of functionality from a single design engineering environment. This goes hand-in-hand with regular reviews to see what their customers are interested in. This happens often via user group meeting, attending fairs, and training events – world-wide. Annual user group meetings are widely used as a means to stay in contact with users.

According to the survey carried out for this report, SWOs interact with their users in a number of ways as shown in Table 1.

Table 1. Ways of interaction with SWOs

Way of Interaction	% of SWOs
user group meeting	12
renewals	15
user interactions	19
papers with software citation	31
feedback	46

Some SWOs come from academia and cater for this audience by providing teaching materials and a lower priced licensing scheme. In some cases, where software was developed as part of national funding, the software has to be provided for free to academics and government institutions. There is also an interest in involving academia to lead a new generation to use the software product.

The size of user communities in materials modelling focussed on the materials (rather than design with the materials) is relatively small in comparison with more engineering focussed modelling. The number of users of mechanical Computer Aided Engineering (MCAE) has been estimated [xi] to be in the region of 750,000 and the number of full-time users of Computational Fluid Dynamics (CFD) at about 300,000 (and up to 500,000 in total including part time users and students) [xii]. Despite their significant size, these figures are seen as still only representing about 10% of the total market size of all scientists and engineers, which is about 30-50 million.

On the other hand, the number of users of electronic Density Functional Theory (DFT), a specific and widely used materials model, is in the 10,000s [xiii].

The survey conducted for this report found that the majority of academic SWOs have over 1000 installations worldwide and often don't know the exact number. All of them can capture downloads from their webpages and repositories and do, but due to unsolicited sources there maybe even more then they expect. The commercial SWO have stricter licensing and can capture their installations more easily. Seven (out of 17) reported more than 1000 installation with the remainder ranging from very few to hundreds as shown in Figure 9.

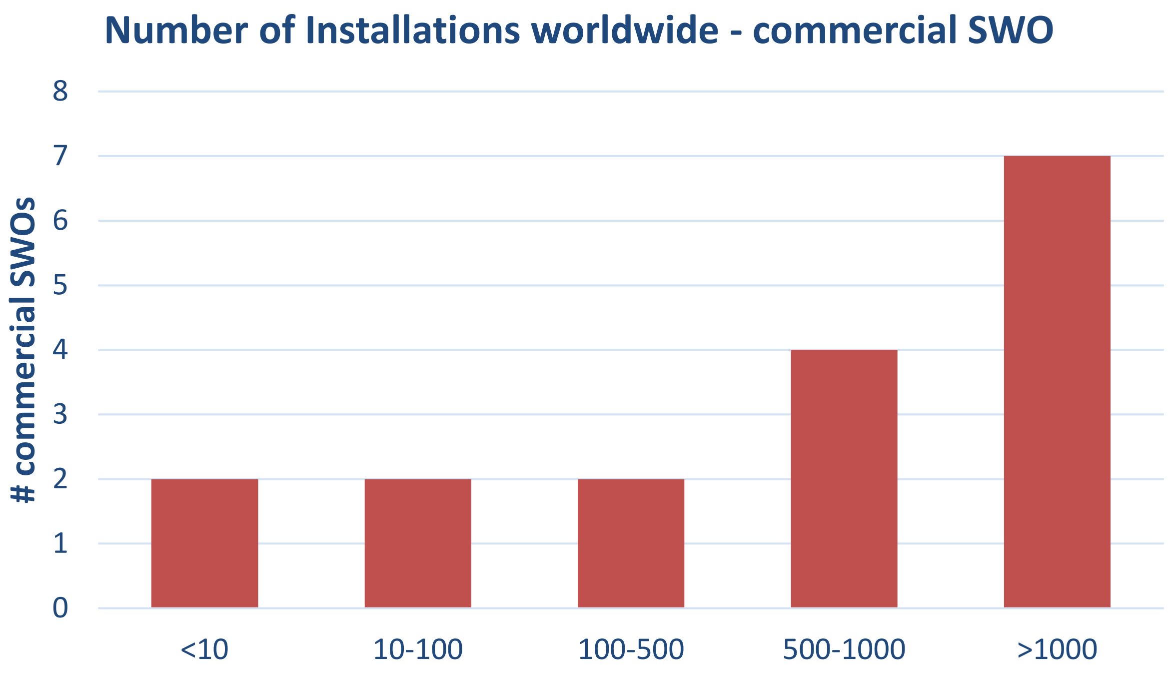

Figure 9. The number of software installations worldwide, commercial SWOs

6.2 Product Management

6.2.1 Value generation

Key to a successful Business Model is a thorough understanding of needs of the market, current problems and gaps and the value generation resulting from the product and/or service.

The SWO is often confronted with the fact that customers may have already a plethora of codes and that their product is used additionally. Thus, many SWOs allow implementation to other codes and if they have GUIs, they make them interoperable, i.e. outputs from competitor/open source codes can be read or they permit generation of input files. Some SWOs would also interface to popular opensource codes. Customers also prefer to get software in one transaction, install it and be ready to run.

According to the SWOs interviewed for this report, the highest values for the users created is that the software let them understand their materials and processes better and if they can reach this understanding very fast. The users want to gain insight into a problem and therefore a good model which is well implemented in the software is needed. Ideally, the software should be robust and reliable. The value of saving costs (i.e. less experiments or similar) and easy to use software seems of a minor value.

Table 2. Important Values from the SWOs' perspective

Value SWO perspective	% of SWOs
easy to use	11
costs	15
robust/reliable/precise	30
insight into problem	30
well implemented theory/functionality	30
understand (new) material/process better	37
Speed/time savings	41

Also, Software users (50% academic 50% commercial) were asked about what value materials modelling software creates for them (Table 3).

Table 3. Important Values form the SWUs' perspective

Value SWU perspective	% of SWUs
easy to use	4
costs	4
well implemented theory/functionality	4
Speed/time saving	15
understand (new) materials/process better	22
robust/reliable/precise	26
insight into problem/enable research	44

Insight into a particular problem and research comes out as the highest value. The persons we asked are involved in R&D and see themselves as researchers and answer the value subjective. The understanding of (new) materials and processes does feature less high as it is a more objective value. The SWUs want their software to be robust, reliable and precise and they wish to trust their results. Users were asked if they have to apply KPIs, i.e. capturing the software value more formally.

Table 4. Formal value judgement by SWUs

KPI with respect to	% of SWUs
Robustness	7
Usage/Utilization	7
Accuracy	11
Quality of support/maintenance from vendor	15
No KPIs	15

Cost as in price/ business model/ licensing	19
Functionality/state of the art /usability/scope	33
Speed/scaling/performance	52

6.2.1.1 Gaps and Barriers to successful software adoption

Replies from academic and commercial SWOs as well as academic and commercial SWUs give some insights into the gaps and barriers regarding materials modelling software adoption, which are summarised below, grouped by people, data and software tools related issues.

People related gaps and barriers include

- **From Academic SWO perspective:** users need to have sufficient education since code was written for expert users, requiring a long learning curve. Also, there is a prejudice of experimentalists
- **From Commercial SWO perspective:** Lack of Modelling expertise and inclination to do experiment instead. User and SWO don't talk the same language – missing/inefficient translation process. Software has often to be sold to manager and not knowledgeable user.
- **From Academic SWU perspective:** need to be expert user, steep learning curve to become a user, lack of support, lack of communication between developers and users.
- **From Commercial SWU perspective:** steep learning curve to become a user, lack of support and training, their peers do not understand why they need to do modelling, lack of communication between developers and users.

Software related gaps and barriers include

- **From Academic SWO perspective:** software is complex, lack of documentation and tutorials, not user friendly, calculations take too much time (and/or may need HPC), no GUI, not interoperable. Also, source code needs compiling and does not easily support OS windows.
- **From Commercial SWO perspective:** Software cannot do what user needs. Complex underlying physics, software is very complex. Calculations take too long. Too low modelling maturity.
- **From Academic SWU perspective:** cost of software, insufficient documentation, not user friendly, too many different codes and GUIs, accuracy, lack of interoperability, no GUI, problem of relating different length and time scales. Also, more IT related issues including that software does not work with Linux. Installation of software can be complicated, need HPC, missing interfaces.
- **From Commercial SWU perspective:** cost of software – often they need more than one SWOs product, insufficient documentation, not user friendly, lack of interoperability, more GPU usage, timelines of SWO and SWU do not match, validation is missing, licensing restrictive. SWOs platform support does not match with what the users have.

Data related gaps and barriers include:

- **From Academic SWO perspective:** Lack of forcefield parameters
- **From Commercial SWO perspective:** User/Vendors cannot provide/disclose sufficient data for modelling or the data is not good enough.

6.2.2 Copyright

According to the UK government (https://www.gov.uk/copyright), "*Copyright protects your work and stops others from using it without your permission. […] You automatically get copyright protection when you create: […] original non-literary written work, such as software, web content and databases.*"
However, due to employment contracts, the copyright is not always owned by the developer as they often undertake software development as part of their job, and in these cases the employer owns the copyright. https://www.legislation.gov.uk/ukpga/1988/48/contents The laws in other countries are often similar to the UK ones mentioned above.
In Figure 10 one can see that only in the case of academic SWOs, developers retain sole or shared ownership of the copyright. In the commercial world all is owned by the company either solely or shared with a third party organisation.

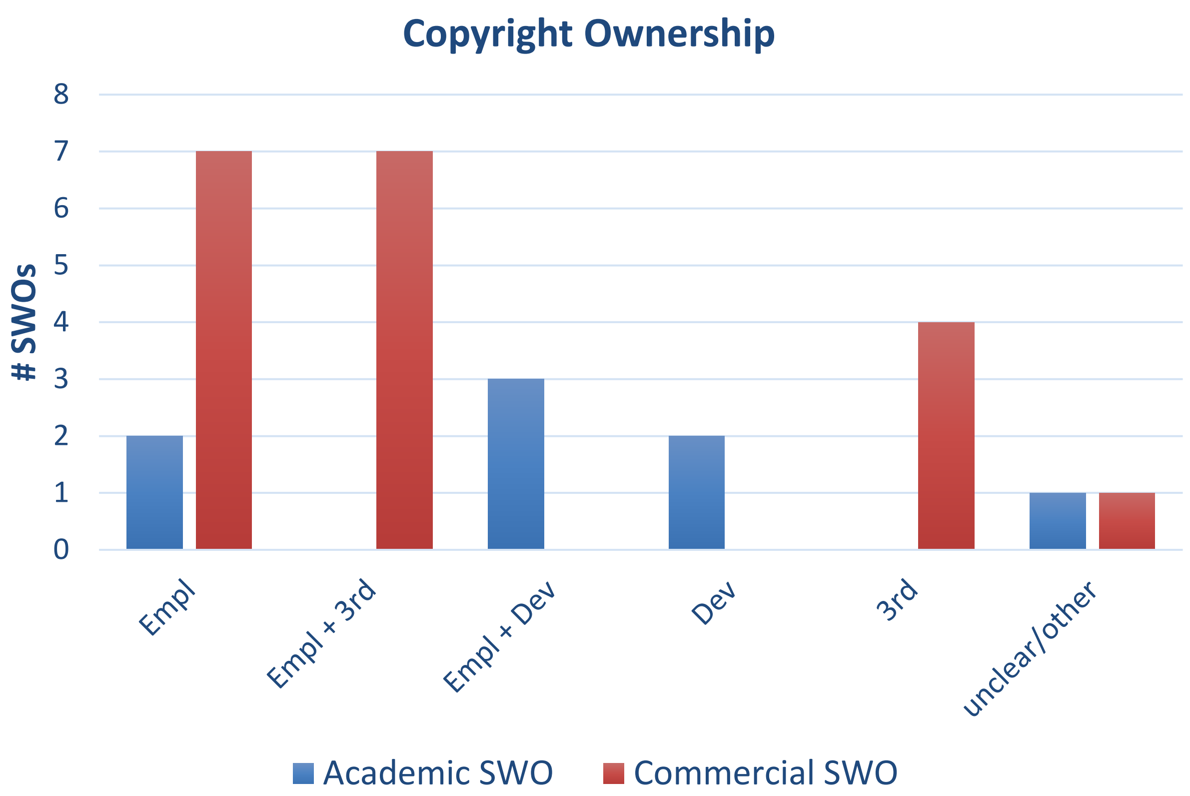

Figure 10. Copyright Ownership: Empl (Employer), 3rd (3rd party employers such as universities, government, other SWOs), Dev (Developer)

6.2.3 Licensing

A software license governs how the software is used or redistributed. The most common licence types are proprietary licenses, copyleft and permissive. All three license types let the copyright owner retain the copyright.
Proprietary licenses permit the right to perform and display. Copyleft and permissive licenses additionally allow the rights to copy and modify the software. For the latter, distribution is only permitted under the same license, which can cause problems for commercial ventures. Only the

permissive license (such as BSD, MIT, Apache) allows a right to sublicense. The copyleft licenses (such as GPL and LGPL) do not; hence, a commercial owner cannot use the software and charge for it. Bespoke Commercial is used to protect intellectual property best, copyleft as it represents the open source philosophy best and permissive if the open source provider allows for better commercial use.

Some academic SWOs opted for a commercial license when they wanted to protect their IP and create income from royalties to fund further development. In this way, their funding and development is not 100% dependent on their host institution and gives them some independence. If the academic SWOs have good support by their funding bodies, they tend to go for copyleft licenses. The commercial SWOs prefer bespoke commercial licenses. If they integrate a 3rd party code, they prefer a permissive license. Some commercial SWOs provide GUIs and other pre- and post-processing capabilities for copyleft licensed codes, for which they can charge. However, the use of a permissive license scheme from the academic environment onwards offers the best guarantee of downstream up-take of the functionality (e.g. by software owners at first, and by industrial end users in a next phase).

Table 5. Detailed usage of bespoke commercial (BC), copyleft (CL) and permissive (PERM)

	Academic SWOs	Commercial SWOs
BC	38%	21%
BC+CL	13%	32%
BC+PERM	0%	32%
BC+CL+PERM	13%	5%
CL+PERM	13%	0%
CL	25%	5%
PERM	0%	0%
not applicable	0%	5%

Additionally, to the licensing the copyright can be permitted for ever, annually, a single user, etc. Academic SWOs who sell their licenses prefer annual or time limited licenses. The time limit coincides with a major release of the software, so 3-year licenses are quite common. The SWOs who have open source codes don't mind how long people use their software so licenses are given perpetually. They prefer single user registration as they can use the number of users for funding applications.

Commercial SWO have more and more the tendency to annual subscription licensing as it allows them to provide package deals of license, support and maintenance. Also, subscription licensing is more easily linked to cloud-based SaaS solutions. Perpetual licenses are therefore becoming less common.

For multiple users, and also if the usage patterns of individual SWUs are difficult to predict the SWO might sell a token software license. The SWU purchases x tokens and if they use y tokens, x-y stay available to other users. These tokens do not deplete, they get fed back into the pool. Also, floating licenses are popular. This has to be run via a license server and users can be authorised and usage can be monitored.

Commercial SWO used to provide perpetual licenses for specific products, but have been moving to subscription tokens that can be applied across their portfolio.

Table 6. The types of Licenses issued

Types of licenses	Academic SWOs	Commercial SWOs
annual or limited to certain time	50%	83%
perpetual	75%	39%
single user	88%	56%
multiuser: (academic) group, university, department	75%	67%
multiple CPU	75%	61%

6.2.4 Software license management

In almost all cases SWO distribute their software via electronic downloads. Access and use it then managed by some license management solution such as FLEXlm[1] or LM-X[2], where FLEXlm seems to be the most widely used. In very few cases Dongles are shipped to the customers to managed the access and use of the software.

The type of permissions and limitations vary and need to be carefully considered. Typically, multi-CPU licenses have become the norm, with limitations on the number of simultaneous jobs.

FLEXlm can be used as back-office as it contains user management tools. This could be used for statistics, to see what feature of software has been used for how long, and by whom. These logs can be analysed and any use can be traced.[3]

“Paper licenses” (trust-based licensing) seemed not to be used in the materials modelling sector, however the principle is seen positively. One can give a company such a license, ask them after a year how many users there were, and then adapt the software offer.

Dongles are rarely used nowadays due to potential complications. They need white listing for a given machine at a given place by IT and may create delays at customs. An advantage for the end user, however, is that it resolves issues that may arise with connecting to a license server.

[1] “FLEXlm software is a prominent license management solution that enables software vendors to impose restrictions on the number of software seats available to their customers. FLEXlm supports different licensing policies such as Floating (aka Concurrent) and Node Locked licenses. This type of software system is also referred to as DRM (Digital Rights Management) Solutions.”

From: https://www.openlm.com/what-is-flexlm-what-is-flexnet-2/

[2] “LM-X License Manager allows software vendors to license their software products to end users in a secure way. LM-X consists of several libraries and tools that can be used by software application developers to protect products against potential overuse, control their license policies externally from applications and enforce various levels of security.” From: https://www.x-formation.com/lm-x-license-manager/

[3] Note however that there may be restrictions due to company rules as well as privacy legislation.

6.3 Software development, maintenance and product life cycle management

6.3.1 From academic code to sustainable software

In almost all cases, materials modelling software has its origins in academia and starts as a code but not necessarily as sophisticated software. It was often coded by PhD students or Postdocs and these tend to focus on their next publication, as is expected in their CV. The long-term maintainability is less of their concern and they are not typically trained in software engineering. Research grants also focus on exciting new science and do not tend to reward writing clean code. Likewise, grant proposal reviewers are typically not concerned with software quality and sustainability, even if it is made a criterion by the funding agency.

It, hence, takes considerable effort on behalf of commercial SWOs to upgrade or rewrite code, and some examples were represented at the Cambridge workshop which took years to complete.

Open source plays of course a big role in academic software developments. A wide range of points were discussed in this regard, including:

- Open source makes distribution easy and facilitates collaboration and wide-spread uptake.
- Open source codes can train people on developing codes.
- Open source development by a community needs strict coordination of contributors. If software developer let other people enhance their code, they may spend up to 40% of their time just doing that.
- Code quality is a problem as open source easily diverges.
- “Leaders” that can keep a community together and ‘gate-keepers’ for quality are important.
- Open source can push the quality of commercial codes.
- Commercial SWOs contribute important testing, bug fixes and are often the only ones that know how to compile/install under Windows and hence get community requests for that: relationship can be mutually beneficial.

Sustainability of a code is vital to generate income for commercial codes, but it is also vital to guarantee survival of an open source code. It was argued that there should be more funding for best practises and software quality. There is a lack in training best practices in software development and HPC. There are still not many university type courses on offer. Even if students take these courses, they can be disappointed because they wanted to do science rather than developing code. Materials modelling codes tend to have a much longer lifetime than typically was originally planned when they were first written. This long-term planning needs to be put into academic developers’ minds.

In the UK there is the Software Sustainability Institute looking after sustainable software. The UK also has an association of Research Software Engineers. On their webpage they state: “*We are an association working to create a community and raise awareness of the UK’s Research Software Engineers. We campaign for the recognition and adoption of the RSE role within academia along with the need for appropriate reward and career opportunities for RSEs.”* There is also ARCHER, a national Supercomputer Centre in Edinburgh, that provides training in best practices for software developers.

6.3.2 Build, buy and partner

Some SWOs mainly develop their own code but also may consider paying royalties if a certain product was suitable to enhance their offerings. They may not write their own code if the 3rd party code is already well used by the community and the final income due to having a better offering may outweigh the royalty costs. Often a commercial SWO may act as the sole reseller of an academic code.

When coding, some developers will work with libraries which are needed to maintain and develop the software. These come with a licensing of their own and the SWO has to make sure all is in line with the licensing of their actual product.

Often databases play a role in materials modelling. These are typically added based on partnerships and royalty-based distribution agreement. The latter is mostly relevant to industry customers as academia may have free access to certain databases anyway.

6.3.3 Maintenance and Life Cycle Management

Software maintenance includes continuous testing/quality assurance during new developments and following each compilation, bug fixes, updates/upgrades of libraries, maintaining current and older OS support while adding new Operating Systems, testing and updating installers, updating documentation and training, etc. All the issues mentioned above are relevant and if not handled well may consume a vast amount of resources. For example, at one SWO typically one person works on just upgrading libraries. Hence controlling the amount of maintenance is an important topic for SWO. See also the discussion about SaaS which would reduce the costs considerably.

25% of the academic SWOs would keep all releases and limit the support for earlier versions. 13% would keep the current version and the one before. For 63% the question is not relevant as either the software is for free and the users can have the newest version anyway or their licensing model would coincide conveniently with a major release and if the SWUs wish to renew they will only get a license for the newest version. About 50% of the commercial SWOs would keep all versions but limit the support often to the current release and the version before. The remaining SWOs have again either licence models to prohibit users to get stuck on older versions, offer web-based services or programme in a modular way.

Also, there is a lack of standardisation of interfaces: SWOs need to maintain too many of these. Despite efforts to agree on common formats etc, it is seen as unlikely that big players in particular will diverge from their proprietary solutions.

Software release cycles are typically annual but for less mature ones they can be 2-3 times per year. Customers don't seem to like maintenance cycles below 1 year as they often have efforts to reinstall new software.

Code tends to grow so at some point it cannot be easily extended anymore. Adding new features and cleaning the code should go hand in hand - maintenance and developing new features become closely fused together. The more dependencies one adds to their software the more care and planning is need for what is needed over the lifecycle. Software often becomes so complex that it has to be rewritten at some point. This is a big challenge in terms of cost.

Customers sometimes request information on tools they used several years ago. Some SWO hence maintain archives including the code and a complied version.

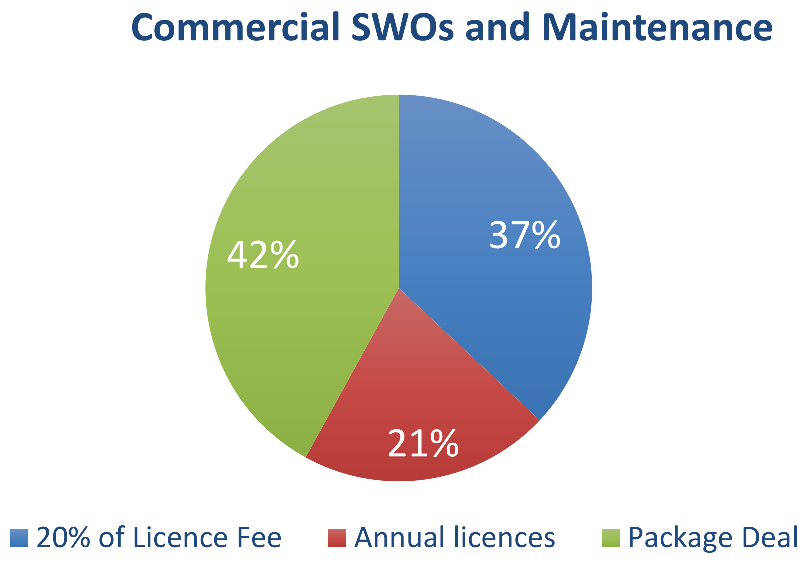

Figure 11. Commercial SWOs and Maintenance

All commercial SWOs (except the 2 SaaS providers) charge in some form. 37 % take 20% of the licence fee and 37% charge special package deals and cannot separate between license and maintenance. A final 21% sells only annual licences where maintenance is included. The interviewed academic SWOs do not charge such fees.

Royalties can also add to costs for SWOs. Academic SWOs hardly have any when it comes to software but may pay for repositories and operating systems. Commercial typically SWOs encounter royalties for the distribution of 3rd party codes and data. Some of these royalties may be one off/flat rate or they can occur every time the software is sold.

The resource requirements to create and maintain the software professionally is significant for both types of SWOs, in particular requiring a number of full-time developers. Commercial SWOs also require a good infrastructure that enables the software to become sustainable. Quality assurance and testing environments are needed. In terms of revenue reinvested into maintenance, academic SWOs often don't have an income that can go back, but if they do, they would invest 75-100% back. Commercial SWOs typically don't share figures, but some mentioned 25%, 50%, 80% and 100% - typically less for established companies and more for start-ups and "young" companies.

The breakdown of time invested into different development activities is shown in Figure 12 and Figure 13 for academic and commercial SWOs, respectively. For the academic SWO we notice some clustering of data with some outliers. The outliers were due to developers making some planned changes, so we may concentrate on the cluster of data to get representative answers. Academic SWOs spend 0-10% on legacy, 5-20% on Operating IT, 10-30% on Maintenance and Bugs and 50-80% on new development. In comparison, Commercial SWOs spend 0-20% on legacy (majority 0-10%), 0-30% on Operating IT (majority 10-25%), 10-45% on Maintenance and Bugs (majority 20-30%) and 25-80% on new development (majority 45-80%). Hence both types of SWOs put most time into New Development, followed by Maintenance/Bugs, Operating IT and Legacy. However, the Operating IT effort tends to be higher in commercial SWOs and new development tends to be higher in academic SWOs.

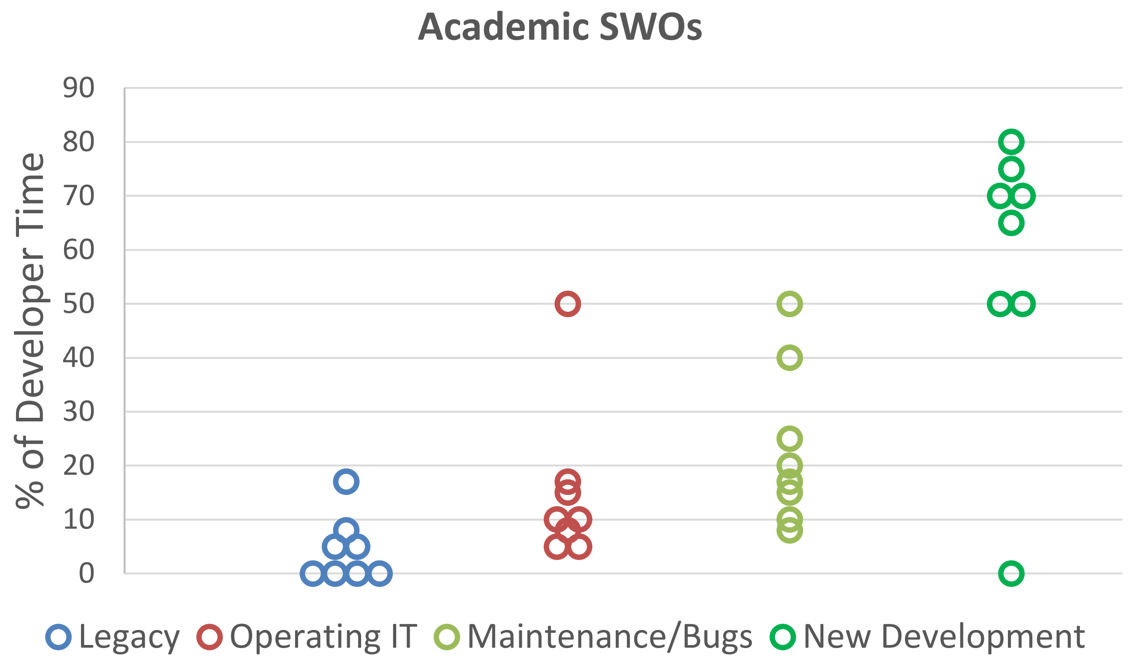

Figure 12. Developer time spent, academic SWOs

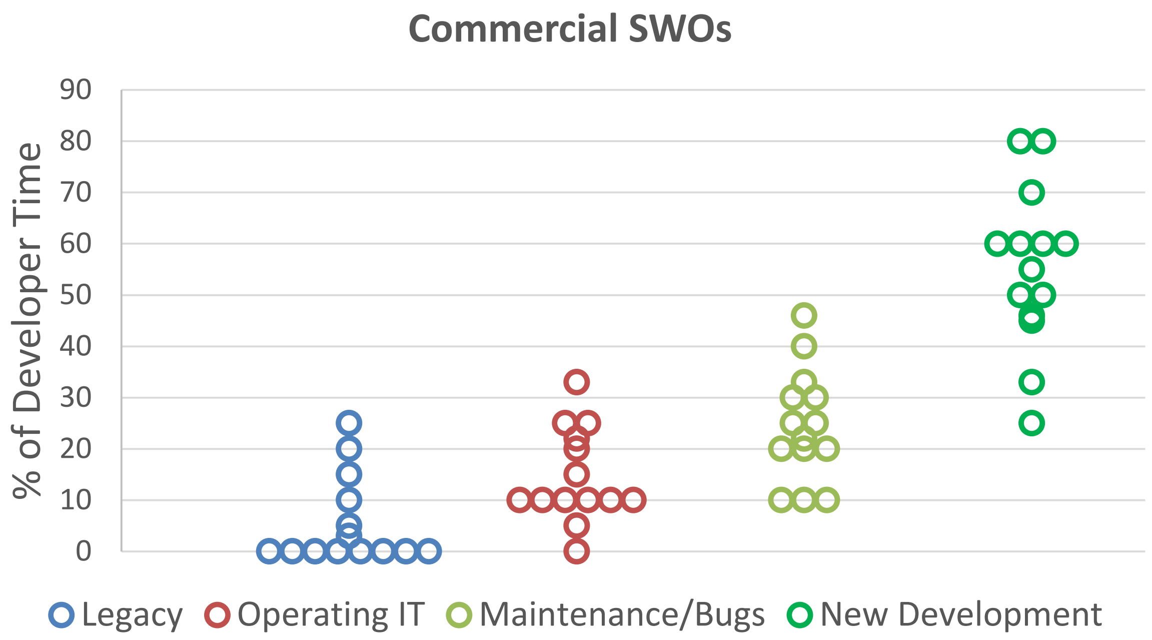

Figure 13. Developer time spent, commercial SWOs

In conclusion, while software is very profitable in principle, since the cost to create one license-worth is more or less the same as it is to create thousands of licenses, it requires substantial investment into its professional development, which can be challenging in particular for organisations with a relatively small number of users.

6.3.4 Software User Expectations

Academic SWUs expect Bug fixes asap if the bug is severe. They also appreciate a workaround for less severe bugs to be provided. If maintenance is charged, they expect a very fast response. New platforms depend on whether the universities have the new OS, e.g. major new Linux distributions. There is a tendency to keep older platforms longer. It is not only the new platform, they also would like to see new compilers. Generally, new releases are expected when there are major new features, as the software has to be installed and paid for. The expectance is here to renew every 3-4 years for commercial software. Academic SWUs prefer additional updates or patches if a new functional or similar comes out before a major release, maybe every 6-12 months.

Commercial SWUs expect bug fixes at least every 3 months and asap if they pay for maintenance. The new releases can cause problems and many commercial SWU do not have admin privileges on their computers and they need IT to install for them. So, once a year is a good frequency. As maintenance is paid for, the expectations for a new release can be between 3 months and 2 years. However, they expect new features and no disruptions to their workflows, i.e. no accuracy changes. New platforms are less relevant as they need new hardware first.

6.4 Revenue Generation

The revenue and business models adopted by the SWOs were captured by the interviews and in the Cambridge Workshop.

Almost all commercial SWOs have business models based either entirely or largely on proprietary software, rather than on open source codes. The exception was one organisation considering starting up a business based on consulting services around an open source code. Nevertheless, open and or free components play a role in the mix of almost all SWOs. There does not seem to be a dominant pattern of mixing proprietary and open/free software however. In particular, some SWOs offer proprietary pre- and post-processing tools (including GUIs) around open source codes while others do the opposite (i.e. offer a free GUI or integration platform for proprietary software). The latter applies in particular where the overall solution for the customer depends not just on one model, but a workflow integrating a range of models.

Also, almost all SWOs run hybrid business models, in particular a combination of software and services. However, it was also pointed out that it can become complex to manage different business models within the same company and some business schools advise against it.

The survey confirmed this picture. 12 of the interviewed commercial SWOs rely on Product Sales and maintenance services, one uses Subscription Based Licensing, 2 rely on services and consulting based on proprietary software solutions, 1 SWO profited from Government funding (assistance awards) and two are looking into Software as a Service (SaaS). One can see a strong tendency towards the Product Sales and maintenance services business model.

Regarding academic SWOs, half of them sell (typically perpetual) licenses for their software, at least for commercial users. Academic SWUs may be able to get software for free or a reduced price. The focus is on software sales rather than services. 90% software to 10% services is seen as the best ratio. The other half of academic SWOs rely on the government funding model, i.e. they get their salary paid for by a university or government facility, but they are expected to raise funds for software development and staff (i.e. PhD students or Post Docs).

6.4.1 Product sales and subscriptions

Product Sales means that the license to use the product is perpetual, but typically maintenance (including support for new or upgraded operating systems and libraries) is included only for one year. Despite a trend towards subscription licensing, it is still widely used (70% of SWO at the workshop).

Revenue is derived from new license sales (i.e. more licenses at current customers, new products at current customers and new customers) as well as maintenance. The maintenance charge is in the region of 15-20% of the price of the software. Most SWO also sell additional services (see below), e.g. comprising training, consultation and implementation.

There are different price points for customers in commercial organisations and academia. The latter is further differentiated by pure research and allowing commercial consultancy.

Subscription based licensing provides access to installed software for a period of time, typically one year. The annual subscription fee includes maintenance. It is also a widely used revenue type, adopted by 60% of the SWOs at the workshop. Note also the wider discussion in the industry and the general trend in the last years to move from perpetual to subscription licenses.

If their licencing permits, some academic SWOs sell their software themselves (or via university) or have arrangements with a 3rd party commercial SWO and let them sell their software. The resulting revenue is not purpose bound and can then be used for salaries and software development.

All SWOs cater for academic users and almost all enable academic use either by not charging for the software at all (12%), giving it for free to academics (19%) or offering it for a reduced price (65%), see Figure 14.

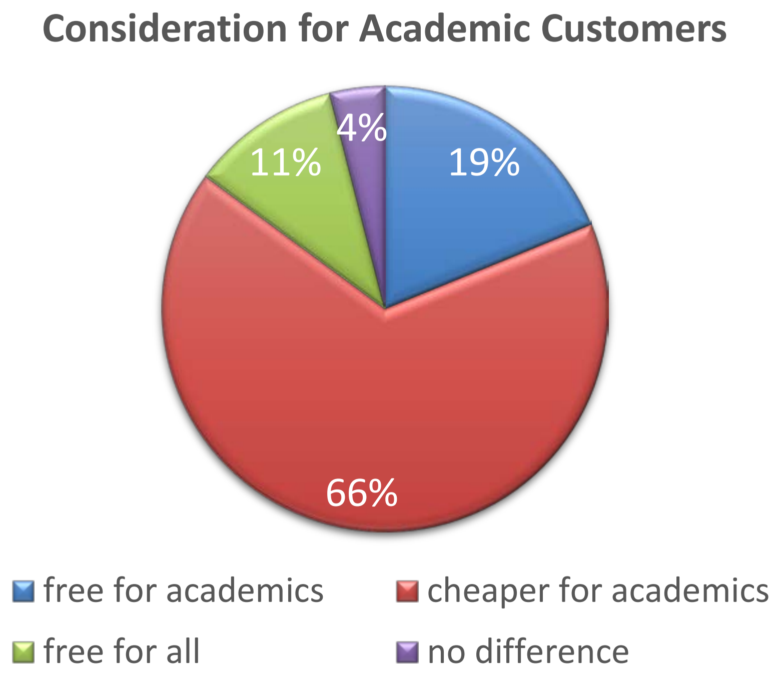

Figure 14. Considerations for academic users by all SWOs

An annual licence for an academic group can be around 800 € for a single software tool to 1800€/annum for a more elaborate software tool with a GUI. This are minimum figures. Generally, SWOs do not publish price lists but have links to sales persons with whom a potential buyer has to negotiate.

We asked SWU what they consider is a typical software budget for one full-time modeller in their type of industry. We used 10-30k € as the lowest budget and especially the academic SWUs pointed out that their budget is rather towards 10k or below. As Figure 15 shows, most budgets fall into the €10-50k range.

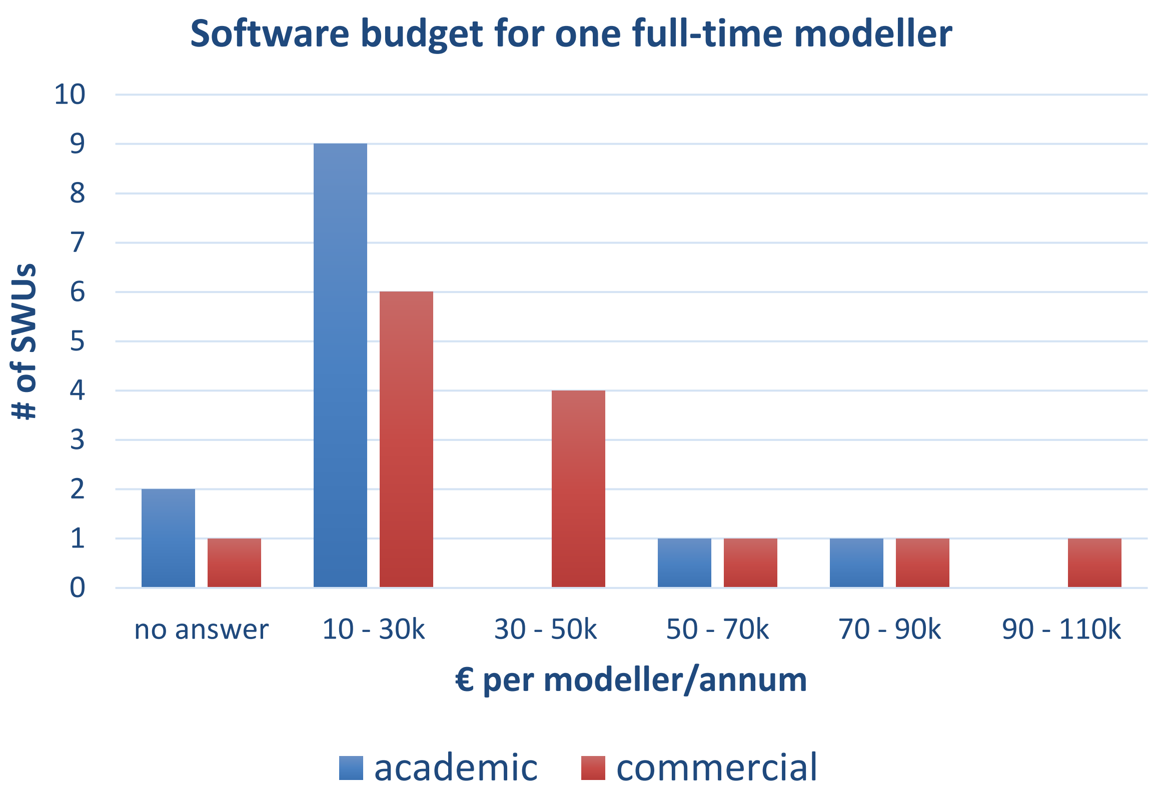

Figure 15. Software budgets of a range of academic and industrial materials modelling users.

6.4.2 Consortium model and Public Funding

Customers Consortia are regarded as very important by several SWOs. The idea is to let the customers define the design requirements for the software, define priorities, test, and provide comments and feedback. Consortia support building and maintain a community and push and pay for development of new functionality. Three of the ten SWOs run customer consortia. None has a consortium with other SWOs. The SWOs involved in these consortia select the consortium members very carefully and give them preferential access to software for a certain time. The value for a consortium customer is that they get the software earlier and can take advantage before it is broadly released.

6.4.3 Services and consulting (based on proprietary or open source software)

The overwhelming majority of SWOs offer services, in the case of the workshop participants it was 7/10. These range from very limited service offerings that are often included in the license fee to implementation, customisation and contract research services. Maturity of the software was also seen as a key factor, with start-ups and less mature solutions requiring a higher percentage of services. Revenue from services makes hence a substantial part of income for materials modelling software.

Table 7. Software: service revenue portions for commercial SWOs participants of the EMMC Workshop

Software vs Service
100% software, no services other than maintenance
30% :70%
Was 0%:100% to start with and targeting 80%: 20%
50%:50%
60%:40%
80%:20%

Table 7 provides a snapshot of across different workshop participants and the portions of their revenue that comes from software vs services.

From a software perspective, reasons for offering services and consulting include:

- Implementation services: very important to make sure the software fits to whatever workflows the company requires.
- The software is very complex and needs consultancy. Consultancy involves setting up simulations and teaching non-expert users how to make the necessary changes for their use cases.
- The huge variety of applications of materials modelling means that the customer may profit from purchasing services that can aid with mastering the software faster to get results.
- Software complexity and maturity (or lack of).
- In depth subject knowledge in the field of the software. The main value in the case of open source software-based services comes from people with knowledge/expertise
- Correlation and updating of models
- Development of new routines on top of the commercial software, either with scripting or with internal routines

In addition, from an end user application perspective, consulting and contract research services are of course offered to provide capabilities and capacities for simulation to industry for projects including:

- Service to combine experimental testing and simulation
- Materials and product development, design and optimisation based on simulations

Caveats regarding services are that (a) they don't scale in the same way as software in terms of revenue and profitability (one can charge people per the hour/day only once, whereas software can be charged several times) and (b) service engagements go through peaks and troughs, which can lead to resourcing issues. One SWO addresses the issue by working with universities that are "centres of excellence" for their software and can provide some services.

While there is no standard with respect to the revenue share generated by services should be offered – the share of services is somewhat related to product complexity as well as maturity of both the product and the market (i.e. user experience), with materials modelling software requiring still a significant amount of supporting services.

The amount of services offered by commercial SWOs varies according to their business models. 100% of services are offered by companies set up as SaaS. They are perfectly content with this. Five of the interviewed SWOs offer 50-70% services and they are not quite content. This offering often stems from the fact that their software is either complex or required the sales of services to push its maturity. These SWOs are looking into increasing their software offer.

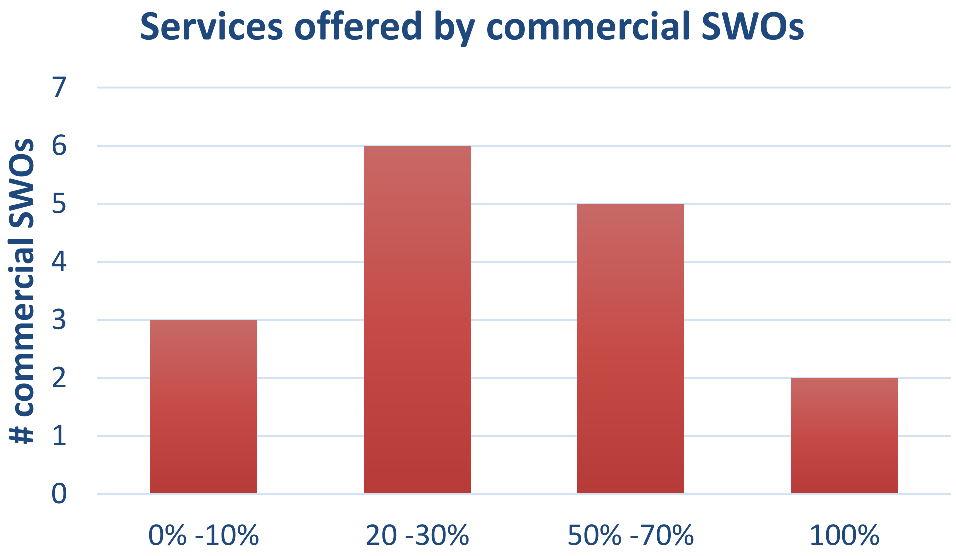

Figure 16. Amounts of services offered by commercial SWOs

Three of the commercial SWOs concentrate on software and offer less than 10% of services. Six of the commercial SWOs who offer 20-30% of services are the most content. When asked, where they see the best ratio in their sector, 20% services and 80% of software was seen as the optimum goal by a majority of the commercial SWOs. The reason is that software can be sold steadily as opposed to a service which is a one-off. However, services are deemed necessary to get visibility with the customer and assure the proper use of their software.

Academic SWOs provide some consulting, support, training examples and online tutorials. All academic SWOs agreed that a certain minimum level is needed to reduce the cases of incorrect use of software. When there is no dedicated staff or funding available, it involves great personal effort. Support is often given via online user groups where advanced users and often developers help. The support is free and voluntarily given so the users have to wait to receive answers to the problems. It works best, when the actual software has reached critical mass with respect to its users. Some users appreciate the help they got with the software and "pay back" to the community by supporting new users. Thus, they established their own learning environment.

Some SWO organise schools, training events for either a small charge or even for free. National Computer Centres who have such software installed may also provide training for local users.

Commercial SWOs provide a whole plethora of services. They all give customers access to training and support. If their software is very complex, implementation and training and support becomes an "all-inclusive software package". Support can range from technical to scientific questions and is a good way to tackle customer issues.

Consulting is often used to prove the software's value, prove a concept/workflow to a customer and to shift the risk for customers. Hereby, the customers can profit from the in-depth software and scientific knowledge of an expert.

Contract Research requires dedicated staff, so is only offered explicitly by a few of the interviewed commercial SWOs. The services can comprise anything from doing research for customers, scripting, contract programming, and customisation.
Implementation Services are offered when the software requires compiling or a particular individual set-up to get started at all. The users get then trained on how to modify the set-up but do not have to start from scratch.
Commercial SWOs were quite interested in collaboration with businesses to extend software capability, drive feature development and validation against specific test systems.
The SWUs are very interested in these services and utilise a range of them.

Table 8. Service used by SWUs

SWU	Installation	Implementation	Customisation	Contract Research	Training
Academic	43 %	64 %	43 %	14 %*	50 %
Commercial	31 %	54 %	38 %	38 %	77 %

* Also includes government facilities

6.4.4 Business models based on open-source and free to use software

"Open Source software is software with source code that anyone can inspect, modify, and enhance."[4] While Open Source does not necessarily mean 'free of charge', we focus here on "open and free" software.
At the workshop, an open-source based business model was represented by an academic group considering a start-up on that basis. The main value comes from people with knowledge/expertise and this is their asset. The aim is to offer consultancy and solve customers' problems. Furthermore, they might charge for software enhancements for a particular customer.
While none of the established companies represented at the workshop had a business model based mainly on open source software, four companies distribute software that includes open source codes. In addition, some companies focus entirely on the commercial software market, while free and/or open source versions of the code are available separately to academics for research purposes.
Reasons for choosing open source include that its visibility makes collaboration easier and enables easier feedback and bug reporting. It is also a "nothing to hide" approach as everybody can look anytime into every line of code. Also, it enables a SWO to have a stake/share in a community around an open source product. It can give the SWO credibility and visibility and could be a sweet spot for open source-based business models.
Revenue is based not only on services, but also on OS-supported and ready-to-install versions, pre- and post-processing tools as well as materials relations (e.g. forcefields) that work with the models encoded in open source software.
In a few cases, software was developed by an end user company for internal purposes and is subsequently made available more widely as open source software. Reasons include that the end

[4] Cited from https://opensource.com/resources/what-open-source

user company is not interested in software revenue, but benefits from making external collaborations with this software easier.

6.4.5 Government funding

Contributions from government funded projects are important to most SWOs, with 7/10 taking part on such projects with funding from national, EU and in some cases US sources. In particular, new developments as well as standardisation and sustainability of software happen often via special funding from the EU or international excellence programmes.

Academic SWOs have positions in universities or government facilities which cover their salary. For some the development of software coincides with their job description. However, it is easier to find funding for doing new science than it is to develop their code further. So, if they want new features in their software they apply for national funding and get a Postdoc or PhD student for the required person hours. Another way is to team up with other research groups and get the needed person power that way. If they want to work on the code as such, they have to apply to international funds such as EU grants. The disadvantage is that the competition is very high (roughly 10% success rate) and the proposals take a lot of effort.

6.4.6 SaaS

Currently, SaaS is not widely used since industry remains sceptical regarding the security aspects and generally doesn’t accept SaaS hosted outside their organisation for business-critical applications and data.

However, SaaS is regarded by SWOs as having the potential to attract customers that do not have the means to get infrastructure and skilled staff in place. For example, it could lead to getting into market niches, especially SMEs, who could not afford a code and the infra structure needed (hardware, skilled personnel, financial investment to get started). SaaS is also as a good way to save cost on software installers, installations and implementations. Also, it is seen as a good way to utilise substantial knowledge around a particular opensource software and to sell simulation services to experimentalists, commanding a premium for best practises/correct workflows.

A barrier for new businesses to emerge is that they may need to license software from other SWOs to offer a specific SaaS solution and there is currently a lack of business and licensing models that would support a win-win situation. For example, one start-up aims to use SaaS to offer workflow solutions to experimentalists, with a usage-based payment model. However, the solution currently requires proprietary codes from other SWOs, hence a licensing arrangement is required. They look for preferential licensing terms from a SWO. In return, they may open up new markets for the SWO.

The interviewees where asked if they offered SaaS and why or why not they may consider it.

Academic SWOs had too few FTEs (full time employees) to consider it at all or their code was not permitted to be offered to industry. Academic users are supposed to handle the software they use to generate data themselves and not go via 3rd parties unless these are seen as collaborator. When discussing SaaS with the academic SWOs, they could see a future for that with small and medium enterprises (SMEs) as they often have not a high modelling maturity, i.e. funds for software, hardware and FTEs. Academic software written under the umbrella of open source and free for all

cannot suddenly be commercialised as it has long-established user communities. SaaS however could be an income stream to co-fund sustainability.
Most commercial providers remain sceptical about SaaS. Their customers do not embrace the idea of cloud computing or running their calculations and sharing data on 3rd party hardware. They have big security concerns and it seems to be the geographical location plays a role. However, there are changes, in the community. A Eurostat Article shows an increasing interest in cloud computing [xiv]. Also, engineering software providers work hard on extending or delivering new cloud-based solutions [xv].
We asked potential users about what they thought about the cloud and HPC as this would be the underlying infrastructure of SaaS. Commercial SWUs seem to have access to HPC and see it as a must or desirable for larger and/or high-throughput calculations. Our interviewees did not shy away from commenting on the cloud and offered positive and negative points. The cloud is used in a few cases and commercial SWUs think it would enable better networking. They prefer to use HPC to get good input parameters and once they have perfect setup, the cloud can be used for production runs. Negatives mentioned were safety and whether or not the cloud is fit for running simulations. The cloud requires knowledge about virtual machines and containers, and comes with upfront high implementation costs. Also, SWUs think the licensing offer of SWOs for the cloud are proving problematic. Generally, commercial SWUs are not against HPC or the cloud if they are the ones in charge of running the calculations themselves and the cloud is private.
SaaS is not quite a topic for academic SWUs as running calculations is part of their job spec. HPC is seen as must and should be a given to allow efficient and complex materials modelling. They agreed that HPC is expensive and that it requires national and international funding to get decent access. Services around HPC are highly appreciated as they would like help with installing software, somebody dealing with technical aspects of a calculation, a provision of a decent queuing system and reliable and stable hardware. The cloud is seen as expensive and controlled by companies which may or may not agree with an individuals' ideology. Also, cloud and security remain an issue.

6.4.7 Data

In addition to software and services, some SWO also sell data that support the modelling, e.g. databases of structures, or thermodynamic data. Also, users are supported by enabling direct connections between models and relevant databases. In exceptional cases, the main function of the SWO is the data, with a range of data-based and some physics-based modelling tools to maximise the value of the offering.
In addition, one SWO interviewed is able to offer validation plus experimental data as well.

6.4.8 Start-up models and experiences

Participants shared experiences with starting up and scaling up a software company. Examples included:

- Establishing a firm base in a specific market segment (such as teaching software and courses) and branching into other segments (industry, starting with specific industry sectors) etc.

- Spin-off with Venture Capital funding: Proprietary software. Caveat that it can go wrong. Organic growth with solid proprietary technology base works better (and led to take over by global player).
- Starting up as vendor of an established and academically widely used code to sell to industry.
- Start-up as a consultancy with high value know-how but relatively immature code. Derive income from consulting to hire software engineers and change from "code" to professional software. Transition to robust software while reducing the consulting/services element.
- Software sales within not-for-profit organisation, i.e. keeping some sort of hybrid between research sector and software sector.
- Consultancy based on open source software, potentially combined with SaaS that solves the issue of installation and configuration
- Very targeted workflows and services, in particular as SaaS for experimentalists. Issues include the need to negotiate license agreements with any proprietary codes that are required for the solution. Currently terms tend to be prohibitive.
- Proprietary pre- and post-processing solutions (GUIs, integration etc) around open source software, funded by investment and organic growth.

6.4.9 Distribution and Sales

The Distribution rights to the software reflect the ownership depicted in Figure 10. As one can see in Table 9, software with academic origin is either sold by the developer and/or their universities. Software with commercial origin or software that was commercialised is mainly sold by the commercial SWO.

Table 9. Distributors of Software:

Distributor	Academic SWO	Commercial SWO
University/Company and Reseller	2	12
University/Company		5
Developer and 3rd party SWO	3	
Free via internet	3	1

SWOs often engage resellers or agents. Academic SWOs do this often to profit from a better market accessibility provided by a commercial SWO and to feed the profit back into their software development. Commercial SWOs take resellers in geographical regions where they don't have an office or where they profit from language skills and local business knowledge.

Pro-active account management and sales are common in materials modelling. The time to convert an initial interest (lead) into sales is often in the range of 6-12 months as shown in Figure 17.

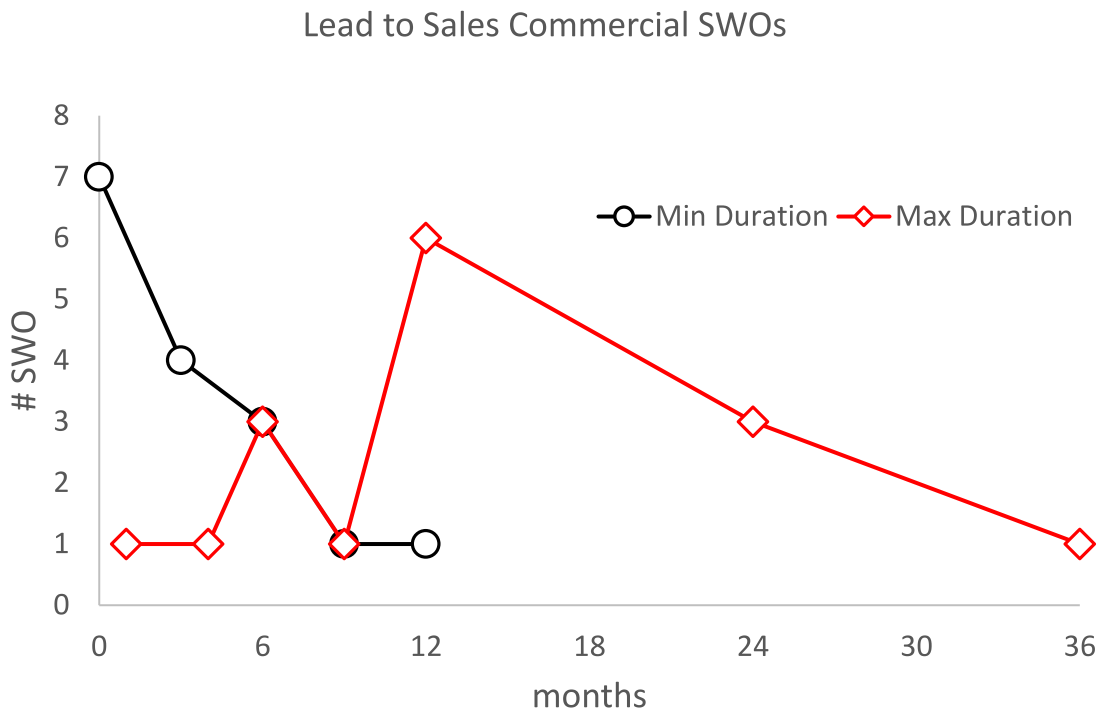

Figure 17. Lead to Sales conversion for Commercial SWOs

It takes a minimum of 0-12 months to convert a lead to sales. A "lead" is a potential customer a sales professional is in contact with and it can take from 1-36 month maximum to achieve a sale. 6, 12 and 24 months were found out to be the most often stated upper barriers. Also, it takes a long time to convert user from one software to another.

For comparison, academic SWO that charge for their software report purchases that take no longer than 3 months to complete.

6.4.10 Growth

There is a wide range of a projected annual growth of commercial SWOs as shown in Figure 18, from low single digits up to 100%. While the largest companies typically lie in the lower range, there was otherwise no specific correlation between company size and growth expectations. Also, there any correlation with model type, i.e. a similar spread for discrete and continuum modelling software businesses.

Several of the smallest companies target in particular growth in staff, whether by revenue or investment.

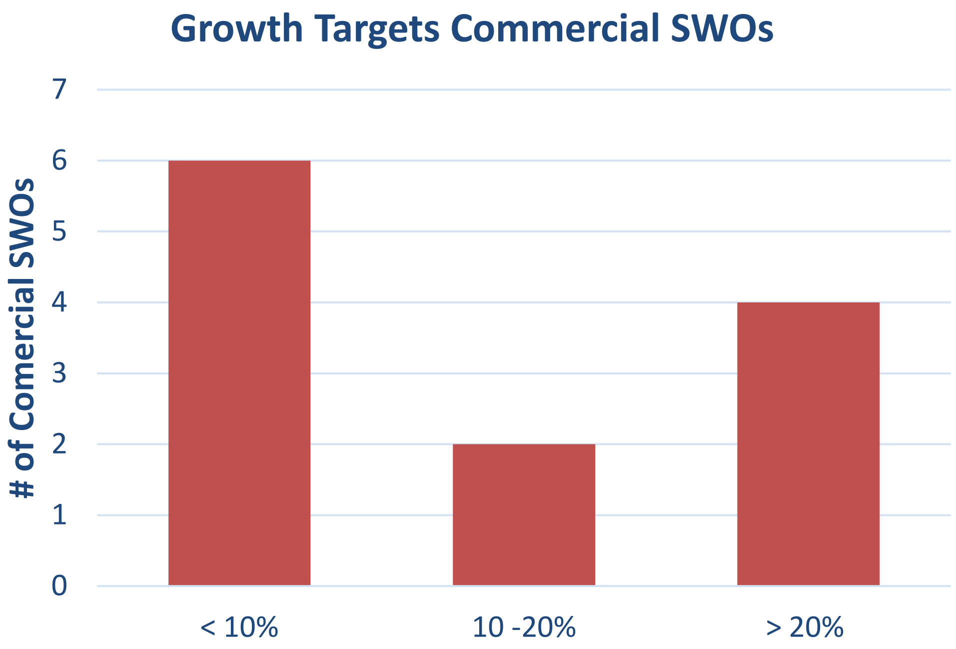

Figure 18. Annual growth targets of commercial SWOs

6.5 SWO thoughts and recommendations

SWOs were given the opportunity to voice their suggestions and needs for sustainable software development.

6.5.1 Academic SWOs

Academic SWOs who can sell their software would recommend that code is written to a high professional standard so it can be integrated smoothly into commercial software. Also, a modular form of the code (usage of libraries and existing tool capabilities) allows easier integration of new functionality.

If the software is mainstream, i.e. it caters for many needs a user may have, it creates more income. However, academic SWUs are very good and well equipped to solve ambitious scientific problems, so one should not neglect specialised codes. To make these viable government funding is vital. Academic SWOs who develop such codes and get funded can establish worldwide renowned research groups and attract international students and researchers.

The academic SWOs mentioned the lack of funding programmes for sustainable software. It is easier for them to get funding for development of a new feature that leads to new science than writing better software. There are some attempts within EU programmes and international funding schemes to provide that but the competition is very high. Thus, academic SWOs would like to see such grants on a national level as well, i.e. sustainable software needs to feature on the national “Road Maps” or in so called “Grand Challenges”.

Software development is seen as a creative process and creativity is best nurtured when it is not under a time constraint. SWOs had the best outcome if they had funding streams secured for many years: the development of a substantial materials modelling code can take 10 years (or more) of development to reach a level of maturity ready for industrial applications.

Another problem that may prevent an academic SWO to deliver sustainable software is the fluctuation of staff. PhD student and post doc effort are a given for a few years and for both publications are more important than the actual software they work on. Academic SWOs need continuity in software development and see this in hiring special staff who are dedicated to the software. This special staff has yet to feature on a university pay scale as professionals. One expects these persons to have substantial scientific AND coding skills, but can only offer a post doc salary and timescale, i.e. short-term appointments. Also, a professional career path and CPD should be offered. It is very fortunate that many disciplines, such as humanities, geography, climate studies, etc. discover research computing and there are movements to make software accessible to everyone, e.g. via https://software-carpentry.org/. The more faculties one can get involved into using and producing sustainable software, the more a University will agree that professional Research Software Engineers are needed.

Academic SWOs who opted for open source did this because they like the open access philosophy and want to provide inclusivity, e.g. accessibility in less economically developed regions of the world. Some governments did like the idea as well and offer financial support. What open source codes are often missing are a GUI, support, and documentation. Often, a community of users take on the role and offer standalone inputs, forums, hands-on examples, etc. Some open source developers try to write their code using permissive licences to enable commercial SWOs to take on their code. SaaS could be of interest as this may create an extra source of income; usually this fails due to the lack of dedicated staff. Often a 3rd party provider may take on this role but will obviously not contribute to an income for the academic SWO.

From the academic SWOs point of view the market is small, highly competitive and very saturated including many open source codes. Established SWOs have a long history in developing their software and don't think starting a venture from scratch would work.

A major rewrite had to happen once the hardware permitted parallel computing. Many of the academic SWOs could do this with 1 FTE. Nowadays, if a new hardware architecture emerges one is looking into a minimum of 4 FTEs to accomplish such change in a decent time frame. Many of the SWOs look for funding to enable the maintenance of a more mature software. The more mature the software, the more hardware/architecture changes one encounters the costlier it becomes to maintain it.

6.5.2 Commercial SWOs

Commercial SWOs charge for their software and would like to emphasise the value of professional software and software engineering. Most of them feel a steady income from software sales and maintenance is key as there is much work involved in converting code into software. It needs large effort and a team of people that require a stable environment to thrive professionally.

For SWOs a well-established market is enticing as they show potential customers how they can offer an alternative solution to an existing one. They embrace new technologies and get customer excited about them. There is also a trend to offer more holistic solutions, i.e. not only predicting a new material for a user, but also provide materials modelling approaches to let them see what they can do with it. All commercial SWOs put effort in having good relations with their customers and follow with interest where industry wants/needs to go. They proactively give their customers new ideas. They acknowledge a need to increase the number of case studies that not only show the scientific advance due to materials modelling but also its technological relevance and business benefits.

SWOs look into making software interoperable to create seamless workflow. Many of them have platform architectures that enable 3rd party or open source software to be linked.

Open source software is seen either as neutral or as positive. Most commercial SWOs are not concerned about losing customers to open source. As stated above the major growth for a commercial SWO is in the industrial market which relies on commercial, well supported, user friendly software with GUIs which lower the big learning curve a customer may encounter when taking on open source software.

Some commercial SWOs see open source as positive as it can foster the field, assist with the creation and growth of communities and markets. Writing open source software is also a good training ground for future commercial developers.

Open source can be embedded into commercial platform solutions, but permissive licensing scheme (such as BSD, MIT or Apache) are preferred as they would allow commercial software exploitation. The license situation is often non-transparent. Commercial SWOs need to know both what license the software is under and who owns the IP.

Commercial SWOs actively seek international funding for tool development. Large EU research projects are favoured with the European SWOs. They encounter close collaboration with industry, work on novel science and are given time to professionally develop new tools. Ideally, they would like to see projects with non-EU industry as well – it would enable them to work with leading manufacturers, who are not always based in the EU.

7. Emerging Business Models: Online platforms and Marketplaces

Despite remaining concerns about 'online' and/or cloud-based systems (see Section 6.4.6), there is an overwhelming trend for software to be deployed and used in cloud-based rather than installed systems as any industry report in this area testifies. Key reasons include:

- It costs less compared to legacy systems
- It cuts operational costs
- It enables companies to maintain a consistent working process
- It enables collaboration in large and decentralized teams
- It saves the time needed for setup and training
- It facilitates access to data
- It is devices independent
- It becomes easier to connect applications with each other
- It reduces (or eliminates) the burden of upgrading, maintenance, and safeguarding data

Likewise, digital marketplaces have been growing rapidly as platforms improving ease of purchasing as well as providing marketing and sales channels for the providers of goods and services. Consumer-oriented (B2C) marketplaces have experienced tremendous growth and many industry reports now analyse and predict an even bigger economy for Business-to-business (B2B) digital marketplaces. Key drivers include improved usability and customisation due to machine learning, the scale, speed and security features of latest cloud platforms.

In the materials modelling field, NanoHub has been an early (open) exchange platform (though not including commercial interactions) which demonstrated the huge growth resulting from online systems. When interactive content and online simulations were introduced in 2005, the number of users quickly grew by about a factor of 10.

However, the development of SaaS based businesses and Digital Marketplace in materials modelling is still in its infancy. In the following, marketplace business models are discussed.

7.1 Marketplace business models and strategies

In principle, one can distinguish between 'one-sided' marketplaces where a marketplace is used as a digital online platform by a vendor, mainly to sell their own product but also to some extent integrate with and provide third party products. Open or two-sided marketplaces are 'bare bones bazaars' that provide a trading platform for providers and purchases (users) to get together and exchange goods and services.

The following marketplace business model clusters are adapted from Ref [**Fehler! Textmarke nicht definiert.**]:

Two-sided marketplace business models:

- Product community
- Offline services on-demand
- Online services marketplace strategies

One-sided marketplace business models:

- Defensive one-stop shop
- Distribution channel extension
- Business model transformation

7.1.1 Product community

Marketplaces help to build a community around products and this type of approach is relevant to digital goods. The marketplace primarily creates value to sellers and buyers by creating an active community of like-minded people. Users of these platforms are, for instance, enthusiastic about discussing and sharing specific content. To foster the community development, firms provide social network functions and focus their key activity on community building. Among revenue streams, commission fees are the dominant revenue form. If fees are differentiated, differentiation is most likely based on quantity. The cluster typically focus on one industry and operates globally. Overall, the 'product community' applies mechanisms and instruments that aim at increasing user engagement with the platform rather than pure transactions.

7.1.2 Offline services on-demand

Such marketplaces match service firms with consumers. The exchanged services are delivered offline and therefore require some form of scheduling. The primary value for both the businesses selling the services and the consumers demanding them can be related to their efficiency gains. For some firms, the value from brand image and platform design play an important role as well. None of the marketplaces creates significant value through the platform community. Rather, these firms focus their activities on generating solutions to increase efficiency. The companies in the cluster generate revenues through commissions from the sellers, while buyers mostly use the marketplace for free. Overall, this cluster resembles the concept of on-demand services. Therefore, we label the cluster as 'offline services on-demand'.

7.1.3 Online services

Companies in this cluster share the characteristic that they offer services that are delivered via the internet. This includes services that involve individuals 'sharing' their previously untapped skills. The cluster also includes firms that aggregate professional freelancers such as scientific researchers. These marketplaces provide a high efficiency to the supply side in earning an additional income or even to substitute their formal employment. Marketplace participation provides sellers with a clear advantage in reaching their target audience. In many cases, the demand-side users receive value from the community around the core service. In 75% of cluster firms, the value proposition is targeted at one single market (vertical). The marketplaces of this cluster monetize by charging sellers (68%), and/or buyers (20%). The fee is mostly charged as commission (55%) or subscription (28%).

From a materials modelling perspective, all three aspects are important for future marketplaces, i.e. community creation, facilitation of services such as Translation and Simulation that may subsequently take place off line and online services.

Also, models may play a role on providing value to marketplaces for other services and goods, such as models that facilitate additive manufacturing trade or potentially models that support the selection of materials for particular manufacturing methods (e.g. models for injection moulding that help with the selection of a polymer for a particular product design).

The following one-sided marketplace strategies are adapted from Ref [**Fehler! Textmarke nicht definiert.**]:

7.1.4 Defensive One–Stop Shop

Provider offers complementary products and services in a Marketplace to build a wide portfolio beyond the initial offering. This strategy leverages the client franchise and aims at increasing value per client by answering needs even outside of the legacy positioning. The digital Marketplace plays a key role to ensure the successful implementation of this strategy. The platform not only facilitates the products and services extension with limited investment.

7.1.5 Distribution Channel Extension

The aim of this strategy is to develop new and complementary distribution channels via the marketplace without jeopardizing relationships with existing distributors and legacy channels.

Three main benefits arise from this strategy. First, the investment required is minimal. Second, it is a first step for the operator to reinforce relationships with distributors. Third, it improves client and business intelligence by providing access to additional data. In this model, the primary challenge is to foster digital channel adoption by clients. To succeed, MP operators need to support their existing distributors while avoiding to be seen as a threat to them.

7.1.6 Business Model Transformation

Players developing this strategy usually see their business at risk either because of upstream or downstream disintermediation or due to strong changes in their ecosystem, such as clients and suppliers going from captive to competitive. By transforming the business model, a player seeks to create value added solutions for both clients and vendors and build a sustainable position. The MP plays a crucial role in this transformation process by accelerating the transition and enabling a more flexible aggregation of services. Also, it provides an infrastructure platform to add new functionalities that enhance client and vendor experience. In order to succeed, the business requires three main elements. First, a successful reposition of its brand. Second, a decisive change management to grow focus on both clients and vendors and integrate service level delivery at the core of the company culture. Third, a swift time to market (TTM) to build a head start anticipating on forthcoming disruptions.

7.2 SWO perspective

Some academic SWOs see marketplaces as a great idea to offer their software. They would want to see a governance and a good infra structure to make these infrastructures viable. However, they cannot invest additional work or carry costs to feature on the marketplace. Other academic SWOs were more reluctant, as some of their code is expert code; they are not sure if such code could thrive without them monitoring it. They also thought Marketplaces are difficult to sustain, are a limited market to limited audience. If there was no uptake by the community, they see them vanishing again. They feel the RoI is difficult to monitor and they are worried that a bad transaction will affect them as a provider.

Also, some commercial SWOs had no or low expectations. They have their own marketing channels in place and are not looking for additional ones. They think a business should have its own ecosystem and a centralised approach is better. Also, if their software requires expertise, training support and not just a slot in the market places, who would take care about their product?

Others could see these infrastructure work if the market places were divided into subsections. If the scale is very big, they cannot see them working. The Cloud and SaaS may cause some problems for users but commercial SWOs could see things like Apps working well.

The more positive commercial SWOs show interest in these market places to sell and advert their software, services, and solutions. If it costs them time and money to join a market place, they will have to investigate whether the RoI is worth it or not.

Barriers:

- EU funds are only for a few years, so there is a worry what will happen afterwards. It takes a long time to build up a critical mass of users, as the NanoHub (www.nanohub.org) example showed. Contrast that with Nanohub (US) receiving 10s of millions of USD funding over more than a decade now.

- Potential customers are sceptical to put data on some site they see as not trustworthy/big/anonymous. They prefer a trusted vendor/SWO.
- Close contact with SWOs is regarded as essential in the materials modelling field in order for the users to derive full value from the code. Marketplace may make the customer interaction more remote, i.e. once further removed than it is today.
- Big companies have their own market places and set their own 'standards'.
- The cloud and running calculations outside of a firewall.
- Market segment in industry that might use marketplaces is regarded as small.

Solutions

- Enable closer contact between SWO and customer.
- VC funding may be available to marketplaces, see ScienceExchange, that has so far received $60m.[5]
- Enable consultants/translators with a good reputation to bridge between the customer and an anonymous market place.
- Facilitate secure interaction and enable strong feedback between users/customers and SWOs.
- Integration plays a big role and new Marketplaces (Marketplace, ViMMP) will cover this.
- Let them be an infrastructure for SaaS and an open, free accessible, well documented exchange structure that anybody can plug into without having the barrier to buy a specific product.
- Find ways to run calculations inside the firewall of a company.
- Focus of marketplaces initially on early adopter market sectors and related models.

Opportunities

- For new codes and start-ups, marketplaces could serve as an intermediate staging place until smaller players become more widely known.
- They can be seen as infrastructure for SaaS and an open, free accessible, well documented exchange structure that anybody can plug into without having the barrier to buy a specific product.

7.3 User perspective

Academic Users

Positive: could be good place for analysis apps, but must be interoperable with their software. Good place for app system, to solve one well-defined specific problem. Useful for academics, particularly if it's funded by somebody else. Good collaboration environment. Will make software findable. Could drive interoperability. Could aid I lowering the barrier to use Materials Modelling.

Negative: "sharing sentiment" will evaporate quickly if a participant can gain a commercial advantage. Some ambitions are so general and large so there is a risk than a too general approach will limit usability. Companies will use it as a one-way street for getting information, but won't trust

[5] https://www.forbes.com/sites/alexkonrad/2017/06/29/science-exchange-takes-on-28m-to-help-companies-outsource-rd/

it with their IP. Most non-theoreticians will still find it too difficult to get started. Is it only for big/established software package to be able to join? Will there be a schism between industry and academics (who even don't have 10K budget !!!and largely depend on free software). Some commercial companies are sometimes reluctant to use such platforms due to security and confidentiality concerns, so this aspect has to be considered very carefully.
Four academic SWUs don't know about it or have no expectations.

Industry Users

Positive: could bring growth to the materials modelling industry. Could serve as a framework to easily select the desired tool. Could aid with facilitation of the flow of information across simulation scales (time and length). Link codes and methodology and aid with workflows. It is an excellent challenge! Make integration and interoperability user-friendly. Good source for case studies and documentation. Could be useful for independent model developers.

Negative: commercial software vendors will not make relevant software available and will not like if there is a platform is created with open source software.

8. Outlook

Materials modelling software lies at the intersection of developments in a number of scientific fields and information technologies. It needs to constantly innovate and incorporate latest breakthrough developments in a wide range of domains and sub-disciplines of chemistry, physics and materials science, engineering and information technologies, including Machine Learning and Artificial Intelligence. At the same time, many legacy applications are expected to be supported for long periods of time by its users. Close interaction with the latest hardware developments and deployment on High Performance Computing platforms is widely expected. Also, the end user profile is gradually changing from subject matter experts to include more occasional users: scientists and engineers but not experts in the specific modelling domain. There is an increasing expectation of software (or software components) from different fields of materials modelling to be integrated, both to support more complex workflows and to provide more integrated information technologies at the enterprise level.

From a software development perspective, there have been many positive developments in terms of the awareness of academic developers of the need for high quality software developments and the management of IP and licensing schemes that allow potential commercial applications and integration. That trend is likely to continue.

Machine learning and Artificial Intelligence developments are likely to impact more and more on materials modelling software businesses as well. There is a need and opportunity for skills development and experts from both the deep scientific fields and data science field to work together. New software developments that combine the strengths of data-based and physics-based modelling could lead to disruptive new software businesses.

There are many factors (as discussed at the beginning of Section 7) that will increase the need for software businesses to support cloud-based and SaaS business models as current concerns are likely to be superseded by the benefits of online deployment. At the same time, the continuous drive and demand to make software components from different sources/vendors more interoperable is very likely to continue. The combination of more cloud-based deployments and increasing interoperability will also support the development of digital online marketplaces in materials modelling. As in other fields, there are very significant efficiency gains to be made by enabling industry to source solutions for their modelling needs via digital marketplaces rather than by interacting separately with a multitude of potential vendors. Likewise, the deployment of software and running of simulations via such platforms can lead to synergies between cloud-based IT systems, HPC hardware, and flexibility of (software and hardware) resource use. Market size and dynamics are of course important factors in such a transition of software business onto marketplaces. The much larger market in the CAE domain (design 'with' materials, see Section 3.2) is likely to be a stronger driver in the first instance. However, it will also be a business opportunity for all types of materials modelling software to access and be integrated into a larger market and for new developments from academia to reach industrial end users more quickly.

9. Acknowledgements

The authors would like to acknowledge the contributions of other members of the EMMC-CSA project to this White Paper.

The authors and the EMMC are grateful to all Software Owners and Software Users who took part in the meetings, the interviews and the online surveys and to their contributions to corrections and improvements of the White Paper. Their effort made this work possible.

The work has received financial support from the EU H2020 project EMMC-CSA GA n. 723867.

10. Glossary of terms and abbreviations

ANDS - Australian National Data Service

BSD – Berkley Software Distribution

CAD – Computer Aided Design

CAE - Computer Aided Engineering

CEN - Comité Européen de Normalisation (i.e. European Committee for Standardization)

CFD – Computational Fluid dynamics

CAGR - Compound Annual Growth Rate

CSA – Coordination and Support Actions

DFT – Density Functional Theory

EMMC – European Materials Modelling Council

EU – European Union

FTE – Fulltime Employee

GNU - a recursive acronym meaning "GNU's not Unix"

GPL - GNU General Public License

GPU - Graphics Processing Unit

GUI – Graphical User Interface

HPC – High Performance Computing

IP – Intellectual Property

LGPL - GNU Lesser General Public License

MCAE - Mechanical Computer Aided Engineering

MIT – Massachusetts Institute of Technology

MP – Marketplace

NeCTAR - National eResearch Collaboration, Tools and Resources

OOPSLA - Object-Oriented Programming, Systems, Languages & Applications, is an annual Association of Computing Machinery (ACM) research conference

OS – Operating system

PLM – Product Lifecycle Management

RDA – Research Data Alliance

RoI – Return on Investment

SaaS – Software as a Service

SME – Small and Medium Enterprise

SWO - Software Owner

SWU – Software User

TTM – Time to Market

ViMMP – Virtual Materials Market Place

VC – Venture Capital

11. References

[i] Goldbeck, Gerhard. The scientific software industry: a general overview. Goldbeck Consulting, 2017. http://doi.org/10.5281/zenodo.260408

[ii] Hettrick, Simon. Research Software Sustainability: Report on a Knowledge Exchange Workshop. Knowledge Exchange Office. 2016.

[iii] Lapham, Mary Ann. Sustaining Software-Intensive Systems. Software Engineering Insitute, Technical Note: CMU/SEI-2006-TN-007, May 2006.

[iv] Benedict, Karl; Best, Mairi; Fyfe, Sue; Habtezion, Senay; Jacobs, Clifford; Michener, William; Nativi, Stefano, Pearlman, Jay; Powers, Lindsay; Turner, Andrew. Sustainable Business Models for Brokering Middleware to support Research Interoperability: A Report from the Sustainable Business Models Team to the Brokering Governance Working Group of the Research Data Alliance (RDA), December 2015 https://www.rd-alliance.org/group/brokering-governance-wg/outcomes/sustainable-business-models-brokering-middleware-support

[v] Raymond, Eric. "The Magic Cauldron," The Cathedral and the Bazaar (Sebastopol, CA: O'Reilly) 1999, http://catb.org/~esr/writings/magic-cauldron/

[vi] Pearlman, Jay and Turner, Andrew. SaaS Business Model, RDA. 2015, accessed via https://www.rd-alliance.org/business-models-draft-saas-discussion.html

[vii] Täuscher, Karl. Business Models in the Digital Economy: An Empirical Classification of Digital Marketplaces. Fraunhofer, 2015.

[viii] Makkonen, Juho. How to Choose the Right Marketplace Business Model. 2015. Accessed via https://www.sharetribe.com/academy/how-to-choose-the-right-business-model-for-your-marketplace/

[ix] de Panafieu, Olivier. THE SPRING OF B2B MARKETPLACES. Roland Berger, 2017.

[x] Osterwalder, Alexander; Pigneur, Yves; Smith, Alan and 470 practitioners from 45 countries. Business Model Generation, 2010.

[xi] Nagy, Dennis. Presentation at ASSESS Congress, 2016. http://www.assessinitiative.com/congress/assess-2016/

[xii] Hanna, R.K.; Weinhold, I. The Demoncritisation of CFD. Mentor Graphics White Paper, 2017.

[xiii] Goldbeck, Gerhard. The economic impact of molecular modelling of chemicals and materials, Goldbeck Consulting, 2012. http://doi.org/10.5281/zenodo.44359

[xiv] Giannakouris, Konstantinos and Smihily, Maria. Cloud computing - statistics on the use by enterprises. 2016. https://ec.europa.eu/eurostat/statistics-explained/index.php/Cloud_computing_-_statistics_on_the_use_by_enterprises

[xv] Schnitger, M. A quick refresher: cloud vs. Subs https://www.linkedin.com/pulse/quick-refresher-cloud-vs-subs-monica-schnitger/, Altair: Bells, clouds, platforms and units bring simulation to the masses https://www.linkedin.com/pulse/altair-bells-clouds-platforms-units-bring-simulation-masses-monica/

Authors	Gerhard Goldbeck, Alexandra Simperler (Goldbeck Consulting Ltd)

EC-Grant Agreement	723867
Project acronym	EMMC-CSA
Project title	European Materials Modelling Council - Network to capitalize on strong European position in materials modelling and to allow industry to reap the benefits
Instrument	CSA
Programme	HORIZON 2020
Client	European Commission
Start date of project	01 September 2016
Duration	36 months

Consortium		
TU WIEN	Technische Universität Wien	Austria
FRAUNHOFER	Fraunhofer Gesellschaft	Germany
GCL	Goldbeck Consulting Limited	United Kingdom
POLITO	Politecnico di Torino	Italy
UU	Uppsala Universitet	Sweden
DOW	Dow Benelux B.V.	Netherlands
EPFL	Ecole Polytechnique Federale de Lausanne	Switzerland
DPI	Dutch Polymer Institute	Netherlands
SINTEF	SINTEF AS	Norway
ACCESS e.V.	ACCESS e.V.	Germany
HZG	Helmholtz-Zentrum Geesthacht Zentrum für Material- und Küstenforschung GMBH	Germany
MDS	Materials Design S.A.R.L	France
Synopsys / QW	Synopsys (former QuantumWise A/S)	Denmark
GRANTA	Granta Design LTD	United Kingdom
UOY	University of York	United Kingdom
SINTEF	SINTEF AS	Norway
UNIBO	ALMA MATER STUDIORUM – UNIVERSITA DI BOLOGNA	Italy
SYNOPSYS	Synopsys Denmark ApS	Denmark

Coordinator – Administrative information	
Project coordinator name	Nadja ADAMOVIC
Project coordinator organization name	TU WIEN
Address	TU WIEN | E366 ISAS | Gusshausstr. 27-29 | 1040 Vienna | Austria
Email	nadja.adamovic@tuwien.ac.at emmc-info@tuwien.ac.at
Project web-sites & other access points	https://emmc.info/