CASE-STUDY: HOW TO IMPLEMENT COLLABORATIVE SOFTWARE SUPPLY- CHAINS – LESSONS LEARNED FROM THE TASK- INITIATIVE

Dieter Hertweck
Daniel Bouché

Heilbronn University
Faculty for Electronic Business, GERMANY
hertweck@hs-heilbronn.de
bouche@hs-heilbronn.de

This paper describes the evolution of organizational structures and collaborative behaviors of forty different software producing companies, in the state of Baden-Württemberg, with a common aim of developing component based business software. The case shows that the role and cultural experience of network facilitators has a major input on further organizational evolution. To analyze the forces for diverse structural evolution and the exploration of suitable implementation processes for regional catalyst are the central intentions of the paper.

1 INTRODUCTION

In July 2004 forty software companies in the state of Baden-Württemberg started a common initiative with the aim of producing standardized high quality component based business software and offering it to regional software-consuming SMEs, in different sectors under a common brand and certified trade mark. The initiative was coordinated by a state agency acting as the regional catalyst with the tasks of project and community management whilst controlling the success.

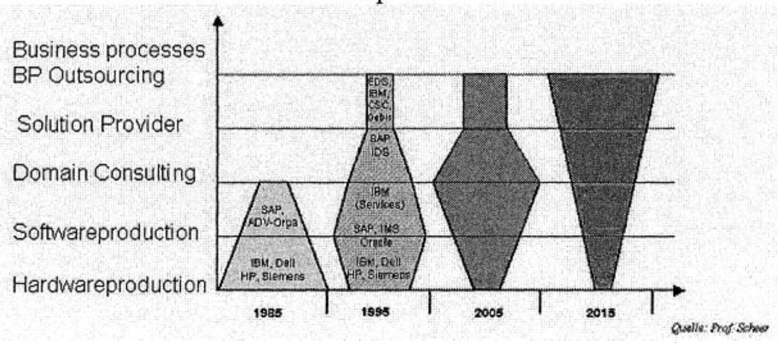
The idea to launch such an initiative came out of a final report considering the future of the IT industry that has been carried out by a work group of IT experts, leading entrepreneurs, government representatives and prime universities from Baden-Württemberg along with the first results of the just started Integrated EU Project Digital Business Ecosystem (DBE). Baden-Württemberg is known for its dense population of software producing and consuming SME, especially in the automotive and machine tool manufacturer sectors. Since their organizational "supply chain" was successful, a major aim of the project was to test the opportunities and boundaries of this approach by adapting it to software production.

Please use the following format when citing this chapter:

Hertweck, D., Bouché, D., 2006, in IFIP International Federation for Information Processing, Volume 224, Network-Centric Collaboration and Supporting Fireworks, eds. Camarinha-Matos, L., Afsarmanesh, H., Ollus, M., (Boston: Springer), pp. 627–634.

A major force in restructuring this scenario of the IT sector was forecast by Scheer (Scheer 2005) and Roland Berger. From their point of view future software and IT service production were based on the following further developments:

1. The coding of software to be developed into a commodity of the future, that could be outsourced to off shore regions like India or Eastern Europe
2. A continuous increase in the development of the value chain of selling hardware and software, towards selling IS solutions, or the execution of whole business processes (see fig. 1). The IS solutions to be customized by the orchestration of existing quality proofed software components
3. The business success of Baden-Württemberg, and of Germany, to be built on the knowledge of business processes, located in smart and medium sized automotive or machine tool companies



Quelle: Prof. Scheer

Figure 1 – future of the IT-Industry

These are the reasons why enterprises in Baden-Württemberg should be able to model and steer the value production in software supply chains, these consisting of: business process outsourcing, IS solution providers and software component suppliers at different levels. This paper shows how the experiences of the project TASK, along with the lessons learned for local governmental units in Europe

2 COLLABORATIVE FORMS IN COMPONENT BASED SOFTWARE PRODUCTION – AN OVERVIEW

Traditional software production is operated by one company with an internal development team with structured software processes, based on models like the Waterfall (Royce, 1970) or Rapid Prototyping (Pomberger, Blaschek 96). The advantage of this production process is the high level of internal quality control and knowledge keeping. The disadvantage is the high number of highly qualified human resources required, for the structured coding of new software releases or versions. To obtain such excellent resources is difficult in a dynamic market. To program software in a stronger object oriented process will help to save human resources. In sum software quality will be gained by the reuse of tested code fragments from the past.

2.1 Open Source (OS) Communities and component orientation

Starting with the disadvantages of traditional programming and the spirit to produce software in a slightly different way, there was a new production paradigm evolving in the late 80's - the open source development often linked with component orientation. The distributed production processes like requirements engineering, coding, versioning and testing were coordinated by the members of the OS community itself, supported by a shared development platform.

An interesting outcome of this production paradigm was the quality of the resulting software being considered as better and with the processes being faster than traditional software production. In the early period of OS, there hadn't been a business model behind the communities. They were often constituted by the engagement of their members and the code of the emerging solution being public, so participants could profit from the working results. With the saturation of good available OS components at the free market like: web-server, security or authorizing components; a new type of small system integration companies appeared. They took OS components as a base for their development of more complex systems like e.g. CRM or ERP. According to the GPL license they didn't charge the software to the end customer but services.

2.2 Component based Software supply chains

In comparison to these loose relationships of idealistic OS-programmers and communities, the supply chain paradigm has a strong division of labor and predefined roles, derived from the future development of the IT sector. The forecast industrial structure soon became reality and was supported by the component strategies introduced by players like IBM (Websphere) or SAP (Netweaver). The degree of flexibility and integration is optimized by implementing cross company business processes using predefined component frameworks. Therefore, software components are produced, integrated and consumed in a vertical supply chain, consisting of infrastructure, middleware and application component suppliers, system integrators and customers. (Vithrana, 2004, see fig. 2).

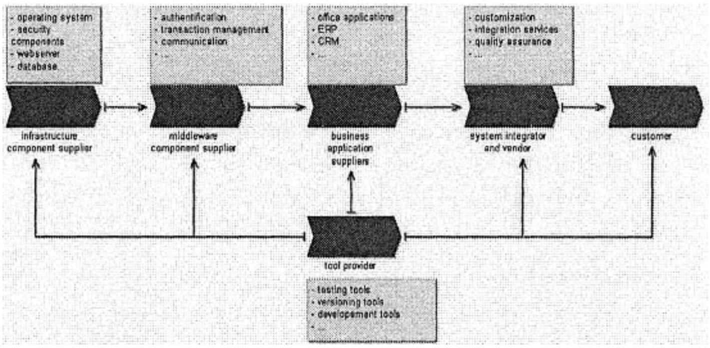


Figure 2 – collaboration model in software supply and demand chains

The system integrator delivers an individual IS derived from the customers' requirements and a fitting IS architecture. The architecture consists of high quality standardized components, offered by suppliers of different tiers (application, middleware and infrastructure) and tested by an independent expert of a quality assurance instance. A shared component development platform delivers helpful tools for orchestration, production, versioning and testing tasks.

2.3 Digital business Ecosystem (DBE)

A digital business ecosystem could be described as a systemic approach including parts of the OS-community and software-supply-chain approach. In difference to them the software and service component platform contains/includes an embedded intelligence, which supports the whole regional network with new business models and tools for self-organization. A DBE consists of:
- Research and education organizations and innovation centres
- Small and large enterprises with their associations
- Local government and public administration can be characterised "*by intelligent software components and services, knowledge transfer, interactive training frameworks and integration of business processes and egovernance models. The latter step in the adoption of Internet-based technologies for business, where the business services and the software components are supported by a pervasive software environment, which shows an evolutionary and self-organising behaviour, will be named digital business ecosystems". (Nachira et al. 2002)*

3 TASK-CASE: IMPLEMENTATION METHOD OF SUPPLY CHAINS IN BADEN-WÜRTTEMBERG

In 2004, when Baden-Württemberg started TASK to establish regional software supply chains, the initiative was embedded in several innovation and research projects, which were quite unique and risky. There was no experience in the transfer of the supply-chain-paradigm to the software sector available. The implementation strategy, chosen by the regional catalyst, was divided in 4 major steps (see fig 3):

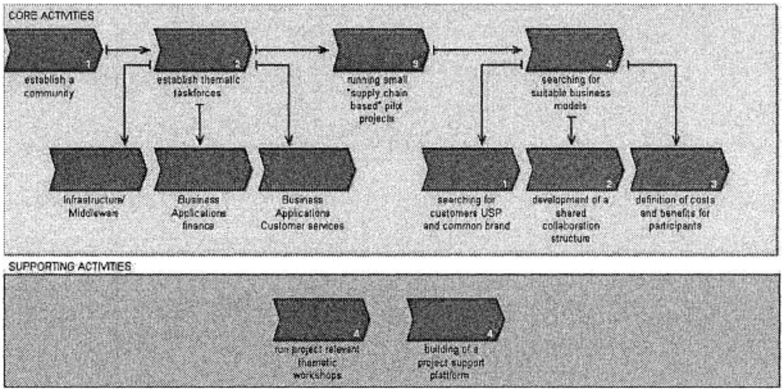

CORE ACTIVITIES
1
establish a community
2
establish thematic taskforces
3
running small "supply chain based" pilot projects
4
searching for suitable business models
Infrastructure/ Middleware
Business Applications finance
Business Applications Customer services
1
searching for customers USP and common brand
2
development of a shared collaboration structure
3
definition of costs and benefits for participants
SUPPORTING ACTIVITIES
4
run project relevant thematic workshops
4
building of a project support platform

Figure 3 – planned implementation strategy

1. Establishing a powerful community of software producing companies
2. Foundation of three different task forces on an infrastructure/middleware and business application level, coordinated by three paid facilitators

3. Running small software-supply-chain pilot projects introduced by the facilitators, while developing a fitting project collaboration and information portal
4. Development of a common business model for all project partners including a common brand, a collaboration structure and a cost-/benefit model.

The establishment of a powerful community of interested software component and service providers was shown that 40 companies joined the TASK community. The strategy of grouping them into three different task forces, to produce component based information systems in a supply chain, was based on the assumption of their capabilities. To instigate the objective, the regional catalyst hired IT experts for each task force, to initiate and support the methodology of the supply chain pilot projects and to implement their management knowledge. In choosing well known moderators - two with long term open source development experience, and one with IT consultancy background - was based on the assumption that they could motivate further companies to participate and donate component solutions at different levels. Furthermore, local events with the topics of collaboration in software supply chains and communities were organized by the catalyst. The overall aim was to gain a critical mass of different reusable software components and the delivery of quality proved systems and services to a potential end customer.

4 RESULTS

Although there was a lot of strategic planning behind TASK, a real project success was the creation of a sustainable software production network; however this wasn't directly visible till the project's end. There was no implementation of a software supply chain in two of three of the pilot projects, because the moderators with an open source background couldn't see the sense behind fixing a collaboration structure as an added value on its own. They succeeded to produce their project results in an OS-community manner. In their opinion the project complexity was so low and the partners so equal to each other in size and power, that a hierarchical supply chain structure would have been a real overhead in comparison to the aimed IS solution.

This was different in the third pilot project that was moderated by an IT-consultant with a business background. He builds up a complete supply chain with a potent system integrator and IT-service provider, as well as different smaller component vendors, into the pilot project. The service providers were especially inspired by the idea of a component supply chain, and the opportunity to market a state own brand to regional medium sized enterprise customers. They forced the whole supply chain to instantiate the customer service solution out of existing software components and defined and developed an interface standard suitable for their project. When the moderator of this task force tried to get a commitment from component suppliers of other task forces and to define a common interface and data exchange standard, he failed. A main reason was that the standardization of a project would have created increased efforts by participants without them seeing the added benefits, by either the potential customer or by the state. This was due to a missing costs/benefit calculation, based on an estimated number of sold software licenses along with the supporting services required. Additionally there hadn't been a clear defined and communicated marketing concept, showing the benefits of supply chain produced component software to end customers and suppliers. Towards the project end, there was a lack of a serious future business model and marketing strategy or commitment to existing models e.g. to the economy of component reusability (Gill 2003).

At this moment the regional catalyst gave us the opportunity to do an independent review on the project to benefit from the lessons learned.

5 LESSONS LEARNED

One lesson learned from this case was the difficulty in implementing regional software supply chains, digital business ecosystem or the type of collaboration networks, without having a clear idea and concept of the future business model.

Traditionally, a business model consists of three elements (Timmers, 98): USP/market position, value chain design and cost/benefit structure. To cement the value chain design right at the start without knowing the product USP for the customer or the costs and benefits for the participants, is a questionable approach.

It would make sense to define the USP of a component based information system by doing a SWOT-analysis first. There would be strengths for the customer like high adaptability and flexibility of the product, as well as benefits for the system integrator, to reach these features by the recombination of existing components cheaply. A weakness of component based information systems is that they don't perform well when fast tracking is required. From this point of view it will be much easier to identify areas where component based information systems have a good USP, like the customer service domain. The next step would be to define the component architecture, with its fitting to the possible customer requirements. In this phase it is mandatory to define a common component interface and data exchange standard for all the participating component suppliers. Without standards, the benefits of reusability and easy integration couldn't be realized. The standardization and architecture definition at the beginning of the project would narrow the necessary variety of success critical components and ensure calculable adaptation efforts later on.

Apparently, if all components were compatible with each other, it would be a question of collaboration culture to establish a suitable organization of the value chain. In supply chains, component suppliers need a higher commitment and should be willing to pay a certain amount of money for services like quality assurance or marketing. Active OS communities are able to deliver continuous work instead of funds. So the question of choosing a supply chain or a community could be depends on the size and investment opportunities of the software companies. Furthermore the power of system integrators is quite immense in supply chains, because they are able to choose only one component supplier the whole time, so that other suppliers have costs for standardizing their components and paying for network services without getting a benefit. The cost and benefit questions are the last major step in successfully implementing virtual enterprise, that should be addressed after defining the value chain design. A component supplier should know how often his component must be sold, e.g. in CRM projects, to get a payback of their investments in the operation of the virtual enterprise.

Confronting the TASK participants with costs and benefits of the successful introduced supply-chain model, a few component suppliers mentioned that they don't need a cost/benefit structure, while others showed interest. It does make sense to structure the project implementation process and strategy differently (see fig. 3). Doing so, the collaboration structure appears as a system variable that will emerge in a self-organizing process similar to the DBE-approach.

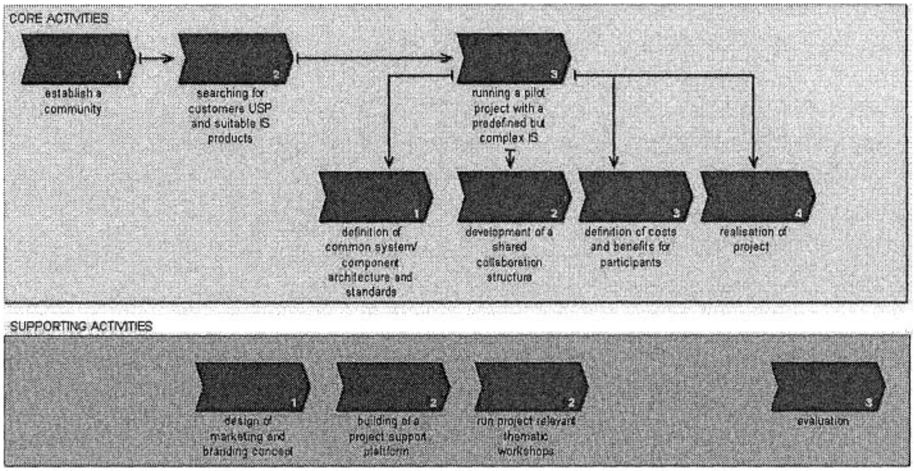

CORE ACTIVITIES
1
establish a community
2
searching for customers USP and suitable IS products
3
running a pilot project with a predefined but complex IS
1
definition of common system/ component architecture and standards
2
development of a shared collaboration structure
3
definition of costs and benefits for participants
4
realisation of project
SUPPORTING ACTIVITIES
1
design of marketing and branding concept
2
building of a project support platform
2
run project relevant thematic workshops
3
evaluation

Figure 3 – target implementation reference process after review of the project

This work should be carried out by all component suppliers, rather than different task forces facilitated by an excellent IT consultant and software architect. The regional catalyst can support these works by increasing the size of the community, along with running marketing activities for a common brand, as well as compensating parts of the standardization efforts the suppliers have to get an easy lock-in (Shapiro, Varian, 1999). The value chain for system production should be co-developed and open to different collaboration scenarios, depending on the customers, integrators and suppliers' economic and cultural structure and their will to take risks or responsibility. Once an appointment made between system integrator/vendor and customer, the integrator can cooperate with his supplier contract or trust based – dependent on his knowledge about the quality of the components and the supplier performance. Acquiring new suppliers consequentially means discussing the issue of potential costs and benefits. Further, a higher number of system integrators in the community will hinder individual system integrators getting too dominant within the community.

6 FORECAST

The TASK case showed the difficulty in establishing a totally new structured virtual enterprise in the software sector. Major lessons learned were the questions of USP and branding, the component architecture, the value chain design, along with the cost and benefit structure, being closely linked to each other in reference to being addressed in the early phase of the project. To develop solely in software supply chains makes no sense if the resulting product, the customer, the authority of system integrators and suppliers are still insufficiently transparent. It was the aim of this case to show a capable method for a regional catalyst of implementing component based software and service production networks. But this was a single scenario. In the future more research needs to be done on business models for virtual enterprises, in the area of value chain design and cost/benefit analysis. There is plenty of practicable literature available about the necessity of trust and social skills (Picot et. Al. 2003) to participate in virtual enterprises or structural factors (Wagner et al., 2004), while less works show benefits and costs of collaborative scenarios in the IT industry. Nevertheless the first results from the DBE project showed (Dini, Rathbone, 2005, p.9), that there were similar experiences made. They identified the following success factors for implementation:

- Global solutions with a local input and sector approach (use of European wide solutions to adapt to regional conditions)
- No single point of failure and control in the virtual enterprise (no predetermination of a centralized organizational structure)
- Commitment to Open Source and Open Standards
- Long-term credibility and attractive brand (work on a consistent brand as the USP)
- Utilisation of proven technologies (reusable component technologies)
- Proven business cases and benefits for service providers and service users.

Bringing these experiences together in developing reference models and best practices, for the implementation of virtual enterprises around Europe, will be both an important and fruitful future issue.

7 REFERENCES

1. Dini, P., Rathbone, N., The Digital Ecosystem Research Vision: 2010 and beyond. http://www.digital-ecosystem.org/Members/aenglishx/linkstofiles/dbe_position_paper_vf.pdf/download, 2005
2. Gill, N. Reusability issues in component-based development, ACM Press, New York, NY, USA, 2003
3. Nachira, F. Towards a Network of Digital Business Ecosystem. Fostering the local development, http://www.digital-ecosystems.org/doc/discussionpaper.pdf, Bruxelles 2002
4. Picot, A. , Reichwald, R., Wiegand, R., Die Grenzenlose Unternehmung, Gabler-Verlag, Wiesbaden, 5. Auflage, p. 123 f., 2003
5. Pomberger, G., Blaschek, G., Software Engineering – Prototyping und objektorientierte Software-Entwicklung, Carl Hanser Verlag München Wien, 1996.
6. Royce, W. W., Managing the development of large software systems', Proc. WESTCON, Ca., USA, 1970
7. Scheer, A.-W., Ein Pionier blickt zurück in die Zukunft. Scheer magazin, IDS Scheer AG, Universität des Saarlands, 2/2005.
8. Shapiro, C., Varian, H. Information Rules, A strategic guide to the network economy, p. 222, Harvard Business School Press, 1998
9. Timmers, P.: Business Models for Electronic Markets. In: EM - Electronic Markets, Vol. 8, No. 2, 07/98
10. Vitharana, P., Risks and challenges of component-based software development. Commun. ACM 46, 8 (Aug. 2003), 67-72.
11. Wagner, C. Europe, Competing: Market Prospects, Business Needs and Technological Trends for Virtual, Smart Organisations in Europe. Report for the European Commission, Information Society 2004.