CompoNex: A Marketplace for Trading Software Components in Immature Markets

Sven Overhage[1] and Peter Thomas[2]

[1]Dept. of Business Information Systems (WI 2),
Augsburg University
Universitätsstraße 16, 86135 Augsburg, Germany
sven.overhage@wiwi.uni-augsburg.de

[2]Dept. of Accounting & Controlling (FRC),
Darmstadt University of Technology
Hochschulstraße 1, 64289 Darmstadt, Germany
thomas@bwl.tu-darmstadt.de

Abstract. Component markets, which facilitate the exchange of components between sellers and buyers, are a key prerequisite for the emergence of reuse-driven component-based software engineering. Subsequently, numerous market forecasts have prophesied component marketplaces to quickly evolve and gain in profitability. However, a long-term observation indicated a rather leisure development until today and proved that only very few marketplaces managed to successfully establish themselves. To investigate possible reasons, this paper analyzes the maturity of today's component markets. Based on the analysis results, it devises critical success factors to counter identified immaturities and provides solutions to transform them into marketplace features. Thereafter, it describes contributions for the architectural design of a model marketplace, which build upon the devised critical success factors and facilitate component trading.

1 Introduction

Mature industrial application development is reuse-driven and utilizes existent software artifacts where possible. It often builds upon component-based software engineering [9], [28], because making use of available components during application development in a systematic manner promises to contribute many improvements – among those a shorter time to market as well as increased application quality, maintainability, and scalability [9], [31].

A key prerequisite for taking advantage of these potential improvements and thus for leveraging component-based software engineering in general is the emergence of global and corporate component markets [31], which provide an efficient access to extensive component stores and coordinate the exchange of components between buyers (i.e. application developers) and sellers. For this reason, reuse-driven component-based software engineering cannot be established without caring about component markets [31] and, since almost ten years, numerous forecasts have prophesied

K. Czarnecki et al. (Eds.): NODe 2003, pp. 145-163

component markets to quickly evolve and rapidly grow both in significance and size [31].

Although both their strategic importance as well as the promising market forecasts should endorse the successful emergence of component marketplaces and one would thus expect component marketplaces to "shoot up" and compete in large numbers, a long-term observation, however, indicated a rather leisure component marketplace development [13]. It proved that most of the marketplaces that evolved until today were unable to establish themselves and rapidly dropped out of sight again. Considering that only very few component marketplaces managed to work profitable, it concluded that the prophesied principal market (r)evolution still did not happen yet.

Starting from this rather contradictive general impression of component markets, in which expectations and observed reality significantly diverge, this paper analyses eligible reasons and deduces a variety of critical success factors that apply in today's component markets. They are taken into consideration as an important strategic basis for the design and implementation of CompoNex (Component Nexus), a model marketplace for trading software components that is currently being developed to face (a variety of) today's difficulties and limitations in component trading and to facilitate component exchange between buyers and sellers under these conditions.

Taking a business perspective, this paper builds upon a general market analysis, which is described in chapter 2, and derives a variety of recommendations referring to the business plan of component marketplaces (chapter 3). The elaborated business plan provides key contributions for the overall technical architecture of CompoNex which are detailed in chapter 4. The paper concludes with an outlook on related work and future research directions.

2 The Maturity of Component Markets

The persisting significant divergence between market forecasts and observed reality indicates the existence of systemic malfunctions in today's component markets which could threaten the profitable management of component marketplaces in general. To get a more detailed impression of the current situation, a structural analysis, which lights up some of the underlying characteristics of today's component markets, is conducted in the following. This analysis will be used to devise a variety of critical success factors for component marketplace operators later on. It assesses the market maturity based on the theoretical assumptions that apply to economically perfect markets [37], [19].

In general, markets can be characterized as special (real or virtual) locations which concentrate on trading economic goods (commodities) to achieve coordination between supply and demand. They manage commercial transactions [17] to achieve good-exchange (coordination) between sellers and buyers and thereby attempt to avoid any kind of friction that interferes with the coordination process [37]. Among several different forms of trading-locations, electronic marketplaces turned out to be a superior platform to trade digital goods like software components, and therefore became a primary trading platform for many e-commerce scenarios [21].

In economic theory, so-called perfect markets turned out to achieve coordination between buyers and sellers in an especially efficient manner. Although it is almost impossible to completely fulfill their theoretical characteristics in practice, even nearly perfect markets (e.g. spot-markets and stock-exchanges) have proven themselves to be superior to other markets and thus emphasized the significance of perfect markets as a generally desirable state.

According to economic theory, perfect markets show the following characteristics [19], [24], [40]:

- Buyers and sellers are competitive (**price takers**) which means that no one has the power to set prices autonomously. This characteristic is usually advanced by the existence of many buyers and sellers, which participate in the market, and the absence of personal preferences (i.e. in general no seller is being favored).
- There are **no barriers to enter or leave** the market, i.e. participants are free to enter or leave the market at any time and without having to incur to special expenses.
- There are **no temporal or regional constraints** regarding the good-exchange. In other words, trading is not affected by the location of participants or market opening hours.
- Offered **goods have substitutes** and buyers view them as interchangeable. In principal, substitutability is supported by establishing standards for the production of goods and avoiding dependencies between goods. Substitutable products ensure that preceding acquisitions of goods have no effect on future acquisitions (discreteness of commercial transactions) and, e.g., changing the manufacturer remains possible for the future.
- Participants have **perfect information** about the price, quality, and overall structure of goods. This information is an important basis for buyers to assess the suitability of goods concerning their individual needs.
- There are **no transaction costs**, i.e. goods can be exchanged without generating additional fees, and good selection, contract negotiation as well as settlement (distribution and compensation) do not cause any costs.

Considering these perfect market characteristics as postulations that also apply for desirable (perfect) component markets, a variety of immaturities in today's markets becomes evident (see figure 1). These immaturities will be identified and discussed in some detail in the remainder of this chapter. In order to endorse the emergence of working markets, it is worthwhile to address the identified immaturities where possible.

Thereby it is principally helpful to differ between custom-made immaturities, which arise from inappropriate marketplace designs, and system-dependent immaturities which inherently arise from the characteristics of the traded good itself (software components) or the current state-of-the-art in the still emerging component-based software engineering discipline. While custom-made immaturities usually can be eliminated by (re-)designing component marketplaces appropriately, system-dependent immaturities in principal have to be accepted as externally predetermined for today's component markets. Nevertheless, in most cases they at least can be mitigated by taking appropriate measures (cf. chapter 3).

An eye-catching characteristic of today's component markets is the fact that many market segments are populated by only a few sellers or dominated by a small number of sellers with outstanding reputation, which contradicts the postulation of price-taking participants. This is especially true for market segments that concentrate on vertical application domains such as enterprise resource planning [13].

The reasons for this situation are twofold. On the one hand, component technologies are still evolving and comprise a variety of shortcomings (e.g. tool support for discovery, configuration etc.) which will not be discussed in detail here. Subsequently, component-based software engineering is still quite a bit away from being a mainstream approach in commercial application development, which limits today's market size(s) and thus the number of sellers. Fortunately, this is expected to change significantly in future (cf. chapter 1). On the other hand, however, especially new as well as small- and medium-sized component sellers usually are not as in the buyers confidence as well-established large-sized sellers, which have a higher reputation and use the resulting power to dominate even more densely populated markets and set component prices.

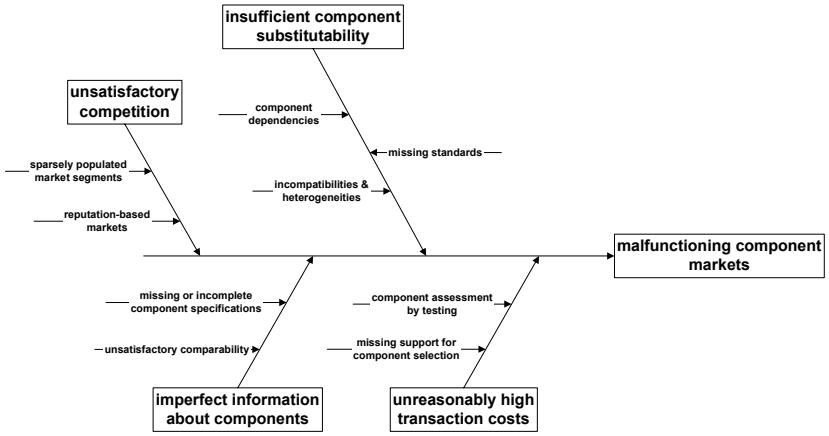

Fig. 1. The cause-and-effect diagram visualizes immaturities in today's component markets.

Today's component markets can thus be characterized as highly reputation-based markets. In such markets, the seller's respective reputation (which usually has to be gained over a longer-lasting period) serves buyers as a substitute for information about goods (especially if offered goods are difficult to assess) and thus plays a key role in good-exchange [39]. Based on the common belief that sellers will not risk loosing their reputation by offering low-quality or faulty goods, a seller's high reputation leads to a high confidence in the goods he offers [30], [39]. Subsequently, goods offered by sellers with high reputation will usually be preferred to those obtainable from less notable or less known sellers.

From an economic perspective, highly reputation-based markets, however, introduce a variety of disadvantages and moreover violate at least two perfect market characteristics: First of all, they contradict the postulation of price-taking participants, since reputation usually is (mis-)used to create personal preferences by buyers and as

a market power to dominate markets by sellers. Furthermore, reputation-based markets introduce severe market entry barriers, since especially new sellers without a competitive initial reputation are usually disadvantaged and therefore unable to successfully participate in the markets. Thus, reputation-based markets also violate the postulation of barrier-free market entry.

Another key characteristic of today's component markets is the emergence of long-lasting trading-relationships between component sellers and buyers. Nowadays, trading-relationships between sellers and buyers usually span multiple component acquisitions, because state-of-the-art components are often dependent on specific components, frameworks, and further (seller-specific) platforms (and consequently are typically incompatible to others) [16]. Thus, (initial) component acquisitions usually bind buyers to the respective component sellers and, in general, components can hardly be viewed as substitutable goods. This, however, violates the postulation of good substitutability and discrete commercial transactions.

Establishing true large-scale component substitutability would require establishing generally accepted technical and content-related (domain-specific) standards for component development. However, current component-based software engineering is far from exhaustive standardization and even exhaustive standardization hardly manages to completely avoid dependencies and heterogeneities between components in practice since standards tend to cement an existing state and are likely to interfere with innovations [31]. In consequence, large-scale component substitutability is a goal which will be rather difficult to achieve even in a longer-term future.

Nevertheless, a variety of standards that focus on smaller market segments and often compete with each other (e.g. in the form of technical component models like Sun's EJB [8] and Microsoft's COM/.NET [12] or domain-specific frameworks like the IBM San Francisco Framework or the Microsoft/Navision Business Framework) are likely to emerge in the nearer future. Following this assumption, component substitutability is likely to gradually advance and improve market maturity [20]. Until component substitutability has achieved a more mature state, however, buyers will have to thoroughly assess the compatibility and dependencies prior to component acquisition.

Unfortunately, buyers nowadays usually are not even able to efficiently assess the usability (i.e. the functional and extra-functional characteristics) of components before buying – to say nothing about the pre-mentioned compatibility and dependency assessment. In order to efficiently assess the suitability of components, buyers require comprehensive information which is typically not provided by today's component marketplaces (and sellers). Thus, the postulation of perfect information is severely violated and buyers are forced to treat components as experimental goods [23], whose suitability-characteristics (e.g. usability, compatibility, and dependencies) cannot be completely assessed until after buying. Being unable to predict and assess component behavior often leads to extreme difficulties in software reuse [16] and makes it very difficult to select between competing products. However, if buyers are unable to discriminate between different goods, the respective markets are likely to malfunction [2].

In order to mitigate this immaturity, component marketplaces today often support the distribution of special component test versions that can be assessed by the respective buyers prior to the acquisition of a fully functional component version. However,

test versions are usually restricted to a reduced functionality and often show different extra-functional characteristics compared to the fully functional version, which makes it difficult to take over assessment results.

Moreover, forcing buyers to gain information (e.g. to compare alternative components) solely on the basis of exhaustive component testing burdens them with significant additional efforts and thus severely violates the postulation of minimal transaction costs. Although component testing may well be an integral part of the suitability assessment process [38], it should not primarily be used to determine component characteristics, but rather to validate them – otherwise, the total assessment costs will quickly exceed any savings that are achieved by component reuse and COTS (commercial-off-the-shelf) based software engineering in general is likely to fail [38]. Besides, while component testing may be able to determine component usability, it is usually unable to correctly determine component compatibility and heterogeneities [16].

Without doubt, ensuring perfect information and minimizing transaction costs are demanding tasks for component marketplace operators, since software components are highly abstract products that always will require a significant amount of explanation and assessment. Thus, it is likely that neither perfect information nor negligible transaction costs are completely reachable goals. However, today's component markets should at least address the mentioned issues and develop instruments to mitigate identified immaturities. Stocks and bonds are impressive examples that even goods, which principally require explanation and assessment, can be traded efficiently – presupposing that mature markets with appropriate measures exist.

3 Critical Success Factors in Component Trading

Especially component marketplace operators are sincerely affected by the prementioned deficiencies in nowadays component markets, since market immaturities have unpleasant consequences for the business success (i.e. component sales, profit etc.). To establish economically successful marketplaces, marketplace operators should therefore address these immaturities already when developing the business plan, which respectively determines the corporate strategy to put a business idea (i.e. a component marketplace) into practice.

Business plan development is usually based on a so-called SWOT (**S**trengths, **W**eaknesses, **O**pportunities, and **T**hreats) analysis [3], which investigates whether a business idea in principal should be put into practice and what precautions ought to be taken when doing so. A SWOT analysis examines the existing market environment and the applying general business conditions to identify strategic opportunities and threats. Thus, the market analysis given in chapter 2 can be viewed as a part of a SWOT analysis to develop a component marketplace business plan (see figure 2). Furthermore, a SWOT analysis is used to determine the respectively available corporate capabilities to realize a business idea and thereby makes own (corporate) strengths as well as weaknesses explicit.

Threats and weaknesses, which have been identified during a SWOT analysis, resemble potential risks which endanger the aspired business success. They have to be

faced by devising critical success factors, which (among others) contain suitable measures to minimize or eliminate business risks [3]. This will be done in the remainder of this chapter. Since developing a complete business plan, however, is highly dependent on the company, which respectively plans to establish a component marketplace, one is principally unable to provide a complete and universally valid corporate strategy.

For that reason, the paper concentrates on devising critical success factors based on the threats that have been identified during the market analysis of chapter 2, which are likely to be relevant in general. To ensure a broad applicability the paper devises critical success factors by firstly elaborating a general strategy and concretizing it with particular measures that may be supplemented with additional (company-specific) actions. The here-introduced particular measures (which are enumerated in brackets) have been developed during a perennial research project and will be included into the technical design of CompoNex in chapter 4, which shall prove their feasibility.

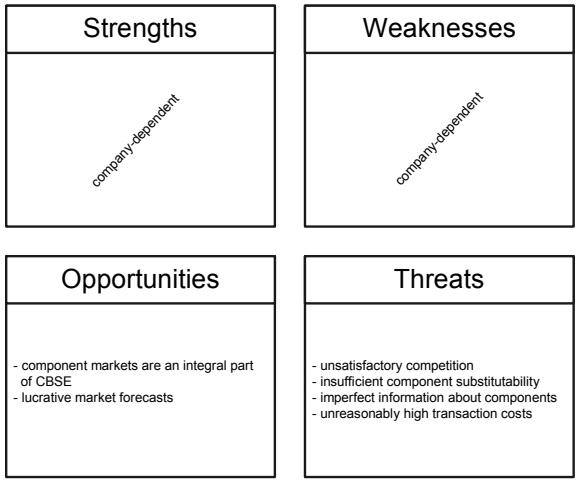

Fig. 2. The SWOT analysis identifies market immaturities as threats.

3.1 Dealing with sparsely populated market segments

When establishing a component marketplace the operators will first of all have to prepare for only sparsely populated market segments, which each are populated by merely a small number of component sellers and buyers. Nowadays, this is especially the case in market segments that focus on vertical application domains. The business strategy of marketplace operators thus in general has to concentrate on ensuring a profitable amount of component trading.

This can be achieved by (1) **covering many component market segments** and avoid marketplace specialization (at least until the situation has changed significantly). On the one hand, universal (all-purpose) component marketplaces are likely

to better leverage investments by aggregating market shares from different segments. Thus, they probably outperform specialized competitors which today have difficulties in gaining a profitable market share.

On the other hand, universal marketplaces need to appropriately support different market segments and target audiences (component buyer groups), which usually differ from one another and have specific demands (e.g. regarding component search, description, support etc.). Therefore, it is essential to (2) **provide a well-structured component catalog** to efficiently access segment-specific components. Moreover, marketplace operators have to pay an increased attention to segment-specific characteristics to avoid deficiencies and attract a critical mass of both component sellers and buyers in each segment.

Since sparsely populated market segments are probably unable to meet the buyer's individual component demands, it is also conceivable to (3) **invert the supplier-inquirer relationship** by permitting buyers to call for tenders. In so doing, component buyers are encouraged to publish their respective requirements (e.g. in form of a component specification) and invite tenders from interested component sellers. The component marketplace delivers these requirements specifications to interested component sellers, who then submit tenders and finally produce the component as ordered (or, respectively, produce a more generalized component to ensure reusability).

In order to increase the trade volume, marketplaces should try to actively match supply and demand. This can be achieved by (4) **providing notification services** that inform component buyers of newly offered components and component sellers of newly published calls for tenders. Moreover, marketplace operators may offer alternative or supplemental components to component buyers.

In addition, marketplace operators should try to attract a greater number of component sellers and buyers to facilitate trading. Component buyers can be attracted by enhancing the marketplace usability by (5) **offering user interfaces specific for particular target audiences** (this is especially crucial for all-purpose marketplaces), (6) **providing community tools**, and (7) **integrating marketplace services** (like component search, browsing, or acquisition) into third party applications to ease market access.

Component sellers can be attracted by reducing existent market entry barriers, e.g. by providing value-added services to (8) **ease the settlement of commercial transactions**. Accordingly, marketplace operators could offer to take over component delivery, solvency checking, or collection. These services not only are especially valuable to small- and medium-sized component sellers but simultaneously open up a new source of income for marketplace operators, which should be able to realize economies of scale and thus can offer these services for competitive prices.

3.2 Coping with reputation as a dominant market power

In order to successfully attract new component sellers, marketplace operators, however, have to address the problems that cohere with reputation-based markets in which component sales are significantly dependent on the seller's reputation (cf. chapter 2). For that reason, the general marketplace business strategy also has to concentrate on

providing mechanisms to increase confidence in the products that are being offered by component sellers with less reputation.

Providing a certification service (9) for both components and sellers [14], which confers the marketplace operator's reputation to component sellers, may especially enhance confidence in components offered by small- and medium-sized or new sellers and thus intensify competition. A certificate is a written guarantee (e.g. a signature, test summary etc.) that a component complies with a given specification of features and behavior [18]. In addition, the conformity to generally-accepted (domain-specific) standards may be attested to provide information about component substitutability.

Additionally, a marketplace operator may (10) **offer escrow services** [10] that facilitate long-term component maintenance by taking precautions against discontinued support or a bankruptcy of component sellers. Marketplace operators may act as fiduciaries and offer to keep component source codes as well as documentation in escrow. If a component seller is unable to continue support, the artifacts in escrow will be handed over to component buyers. Although escrow services are unable to avoid discontinued support, they nevertheless mitigate the risk of acquiring third-party components and thus are able to enhance confidence especially in small- and medium-sized component sellers [10].

Finally, providing means that (6) **enable buyers to provide a feedback** on (the performance of) component sellers and previously acquired components facilitates the discrimination between components without regard to the seller's reputation by enhancing confidence in high-quality products and marking faulty components.

3.3 Providing appropriate information about components

Finally, the marketplace business strategy has to include measures to remedy the problems of insufficient information and unacceptably high transaction costs, which mainly originate from exhaustive component testing. In other words, marketplace operators (in cooperation with component sellers) should provide the information necessary to efficiently discriminate between components and assess their suitability.

This can be achieved by making available (11) **appropriate and automatically verifiable component specifications**. Unlike test versions, component specifications explicitly describe component characteristics and thus can directly be utilized by component buyers. By evaluating a component specification instead of a test version the overall transaction costs may be significantly reduced. The extent (what is to be specified) and formats of specifications (which notations have to be used) are determined by a specification framework that, accordingly, has to be developed.

To support discrimination between components as well as component assessment, a specification framework should include information referring to acquisition (e.g. the price, license agreements etc.) and technical aspects. Technical specifications ought to contain the component dependencies as well as the standards to which a component complies. This information is helpful to determine the component substitutability and the compatibility with other components. In addition, they should completely describe the functional and extra-functional component characteristics. This information is helpful to assess the usability of components.

Establishing a specification framework not only facilitates discrimination between components and component assessment, but also is an important prerequisite for offering certification and call-for-tenders services, which respectively require component specifications as input. This emphasizes the strategic importance of component specifications as an instrument to facilitate component trading. At the same time, the usage of specifications to provide information about components is likely to intensify the demand for certification, which guarantees the factual conformity between specification and implementation (and thus renders expensive validation superfluous).

4 CompoNex: A Model Marketplace

The identified critical success factors of the business plan provide an important methodical basis for the design of component marketplaces. They will be taken into consideration during the implementation of CompoNex (Component Nexus), a model marketplace that is currently being developed to practically evaluate proposed measures, which promise to overcome (or at least mitigate) known immaturities of today's component markets and to facilitate component trading. It is a part of an extensive methodology, which is under development and (in the long-term) aims at completely supporting commercial component-based software engineering using a mix of specialized cooperative methods and tools.

The CompoNex development project was started in April 2003 after the launch of an initial prototype, which has been part of a feasibility study. CompoNex is being designed as a publicly accessible, universal (all-purpose) component marketplace (as postulated within the business model of the previous chapter (1)). It concentrates on facilitating the exchange of components between software developers and thereby implements a commission-based profit policy. Its current stage of development, which will successively be augmented with newly developed features, is available under http://www.componex.biz.

Ongoing development efforts concentrate on designing and implementing a productive and modular marketplace version. The marketplace design builds upon the critical success factors given in chapter 3, which contribute key parts of the overall architecture. Thereby, core features that principally have to be implemented to enable component trading and value-added features that facilitate component-trading are distinguished. While core features are given priority, value-added features will successively be implemented on top of the core features. Currently, the implementation of core features is completed, whereas most of the value-added features are on the drawing board – however, some general impressions on their design will nevertheless be given in the remainder of this chapter.

4.1 Architectural Design

The architectural design of CompoNex currently comprises features that are derived from the identified critical success factors (cf. chapter 3) and have respectively been modeled as (architectural) components. In principal, the design has been based on four essential principles:

- It has to be **modular** in order to facilitate marketplace maintenance, scalability, and feature extensibility [31]. To guarantee an appropriate implementation support for the emerging modular architecture, CompoNex builds upon the component-based Microsoft .NET framework [12].
- It should **reuse existent functionality** where possible to reduce the overall development costs, increase implementation quality, and shorten the time to market [31].
- It ought to **encapsulate individual features** to enable their autonomous practical evaluation ("unit evaluation"), independent evolution, and plug-and-play feature composition in order to determine an optimal feature-mix ("configuration evaluation").
- It has to **make features accessible** for (third-party) clients in order to facilitate the integration of CompoNex into existent development environments and thus to enhance tool-support for component-based software engineering in general (critical success factor (7)).

The marketplace comprises an architecture that complies with these design principles and introduces three conceptual layers (see figure 3): the implementation layer contains the functionality and is divided into several components (each encapsulating a marketplace feature). The service layer provides a common interface which exposes the implemented functionality to any clients that access the marketplace. This eases the development of different clients (e.g. specialized to target audiences as postulated by (5) within the business model) as well as the integration of CompoNex into third party applications (e.g. CASE tools).

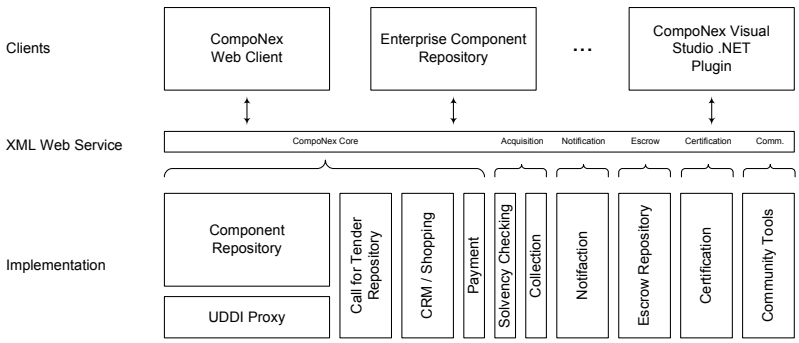

Fig. 3. The overall architecture of CompoNex is divided into marketplace Implementation, services, and clients.

To provide a platform-independent access to marketplace features, CompoNex exposes an application programming interface (API) based on the XML Web service technology [6]. Marketplace features may be invoked using the Web service interface, which is available at http://www.componex.biz/webservice/componex.asmx and provides core-services as well as value-added services (see figure 3).

Currently, two clients have been developed as parts of the CompoNex client layer (see figure 4): a Web based graphical user interface (GUI) that is available at

http://www.componex.biz and a (prototypic) plug-in to integrate marketplace services into Microsoft's Visual Studio .NET.

The core features comprise a repository to store components, a repository to store calls for tenders (success factor (3)), and the marketplace customer relationship management/payment. The component and the call-for-tenders repository implement a specification framework and a component model to provide a structured catalog (success factor (2)) that discriminates several component types and application domains. Moreover, both repositories use the specification framework to offer a detail-search mechanism to find components and calls for tenders, respectively, based on a reference specification. Thereby, components similar to the search query are included into the result-set and rated appropriately. The specification framework determines the structure of the provided component catalog and influences the design of many of the marketplace features. Since it moreover has been recognized as a critical success factor (11), choosing an adequate one becomes crucial to the marketplace success in general and will be discussed in detail later on.

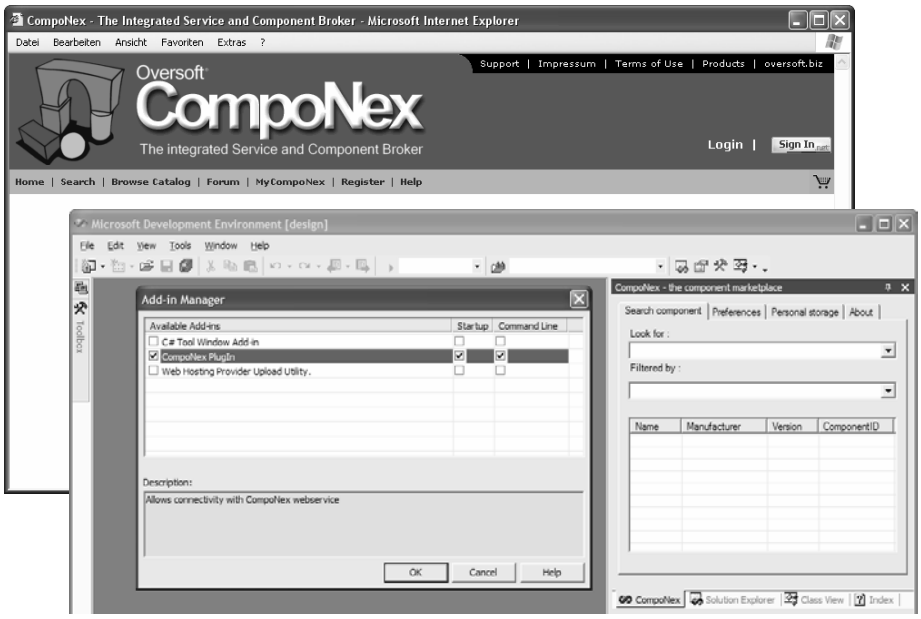

Fig. 4. Screenshots of the implemented CompoNex clients (Web based user interface as well as Visual Studio .NET plug-in).

Within the core features, the customer relationship management is used to store contact information and to log commercial transactions. Based on this data, the payment component is used to collect fees from component sellers accordingly. The payment component will later on be augmented when implementing the value-added services. Then, it will also be used to receive payments on behalf of the component sellers and interoperate with the marketplace services that realize solvency checking and collection (success factor (8)). Simultaneously, the component repository will be augmented to support automated component delivery.

In order to actively match supply and demand (factor (4) within the business model), the marketplace contains a notification service, which belongs to the value-added services. This notification service uses a publish/subscribe pattern [15] to notify component buyers of newly offered components and suppliers of published calls for tenders. To be notified, buyers and sellers can register their respective search query to the notification service, which performs it in regular intervals and automatically delivers the results.

In addition, escrow and certification services are being offered to increase the trust in third party (COTS) components and thus to facilitate component exchange (which has been postulated by (10) and (9)). The escrow repository stores component source codes and documentation (i.e. white-box models focusing on the component internal design, test cases to test the internal design etc.), which will be made available to component buyers if the respective seller declares bankruptcy or a customized condition is being signaled (e.g. if the support is discontinued).

The certification service builds upon the component specification framework and takes a component and with its specification as input. Both are used to validate the compliance between a component specification and its implementation, which is certified by a digital signature.

The offered value-added services are complemented by the CompoNex community tools, which provide a forum for the exchange of feedback on components, reference configurations, and the performance of component sellers. The provided user forum is being designed to further improve the marketplace usability as postulated in (6).

4.2 Specification Framework and Component Model

A variety of marketplace features build upon a conceptual component model (providing component classifications) and a standardized specification framework that delivers information referring to offered components or published calls for tenders (cf. chapter 4.1). Thus, the implemented specification framework, on the one hand, becomes a crucial instrument to integrate (and harmonize) the different marketplace features within a homogeneous methodology.

Since the specification framework, on the other hand, also pre-determines the amount of information about components that can be utilized by component buyers/sellers and marketplace features, it has to be designed to support a variety of different tasks (e.g. component identification, certification, evaluation, acquisition, and interoperability tests). In order to comply with these requirements (which also have been recognized as a critical success factor (11)), the CompoNex specification framework is based on a standardized specification framework for business components [1], which it augments to specify components of arbitrary domains. The resulting specification framework is described in detail in [26].

It builds upon the structure of UDDI [32], [33], [34] and provides thematically grouped information referring to components using a variety of perspectives (see figure 5). Moreover, it uses a mix of well-established notations to denote specifications in order to guarantee an appropriate tool support and thus to ensure automatic verifiability (success factor (11) cf. chapter 3). The CompoNex specification framework supports the specification of the external view of components only, i.e. it pro-

vides information about what a component is doing without detailing how this is achieved. Thus, the component specifications provided by CompoNex do not contain the business-critical internal (white-box) view on components (which is a valuable result of the development process that is usually kept secret by the respective component sellers).

The specification framework contains general and commercial information (e.g. about the component seller, distribution channels, pricing, scope of supply etc.) within the so-called white pages.

The yellow pages contain component classifications according to the CompoNex component model (e.g. the application domain, the architectural component type etc.). These classifications are used to efficiently store and retrieve components [27] as well as to develop structured component catalogs (which was postulated by (2) within the business model cf. chapter 3).

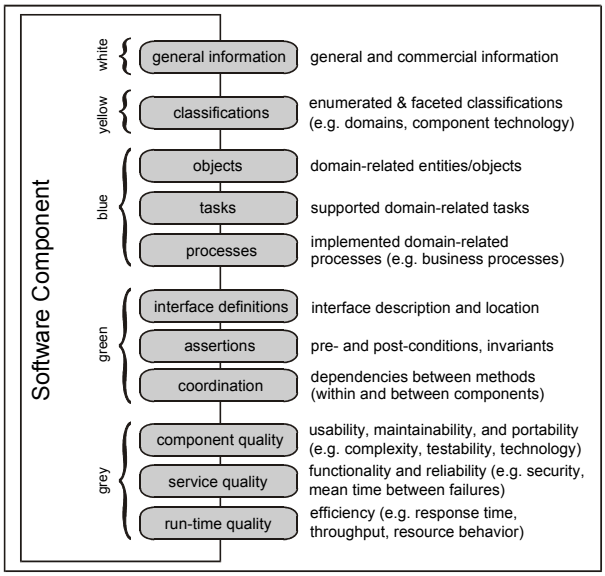

Fig. 5. The CompoNex specification framework provides information referring to different component perspectives.

The main part of component specifications, however, is devoted to provide technical information about components in order to support compatibility checks and automated assembly (see figure 6). Thereby, the blue pages characterize the component functionality by describing domain-specific concepts that have been used for component development (the so-called domain model [11]). They contain the conceptual information model, the operations model, and relevant conceptual processes (e.g. business processes). The proposed notation for the specification of concepts builds on a lightweight ontology and is similar to natural language which enhances the readability of concept definitions.

The green pages contribute information referring to the component structure, i.e. its required and provided interfaces. They comprise the respective interface definitions,

assertions [22], and the component protocol (in the form of coordination constraints). To denote the component structure, the usage of OMG IDL and OCL is proposed as principal notation.

The grey pages expose information about extra-functional properties and describe the component technology as well as the component quality attributes according to the ISO 9126 quality model (i.e. usability, maintainability, portability, functionality, reliability, and efficiency). The primary notation to describe extra-functional properties is QML (cf. [26]).

In total, the CompoNex specification framework introduces five perspectives on components that are subdivided into several specification aspects. This leads to a significant amount of specifications that have to be delivered by the respective component seller and raises the question, whether they will agree to provide all of the specifications that have been proposed above. While this concern is difficult to answer in general, it is nevertheless conceivable that component sellers will be able to provide the requested specifications, since most of them are produced as by-products during component development (blue pages contain a part of the domain model, green pages contain a part of architectural design, and grey pages contain some of the test results).

	Functionality / Concepts (domain-related)	External Structure / Interfaces (logical)	Implementation / Quality (physical)
Static View	Objects (Data Model)	Data-Types	Usability, Maintainability, Portability
Operations View	Tasks (Operations Model)	Events, Methods, Assertions	Functionality, Reliability
Dynamic View	Processes (Flow Model)	Coordination Constraints	Efficiency

Fig. 6. The CompoNex specification framework introduces three perspectives to provide technical information about components: functionality (blue pages), structure (green pages), and implementation (grey pages).

Moreover, CompoNex does not force component sellers to provide all of the proposed specifications (except for using the certification service) to participate in the market (therefore, in fact, specifying the white and yellow pages already is sufficient). However, providing detailed information about components is an important competitive advantage for component sellers, since it allows potential buyers to better assess offered components and thus increases their confidentiality (cf. chapter 2). Thus, the already discussed market mechanisms are likely to encourage the availability of "complete" specifications.

The CompoNex specification framework is complemented by a conceptual component model, which has been designed to systematically structure the catalogs that contain offered components and published calls for tenders in order to facilitate their retrieval and the development of different views for clients specific to certain target groups (success factors (2) and (5)). It classifies components according to the implemented functionality and the reuse models used to sell components.

To classify the component functionality, the CompoNex component model firstly distinguishes between architectural component types [1]: components (in a narrow

sense), frameworks, and applications. Components concentrate on providing specialized services and (partially) automate the execution of a particular task [31]. Frameworks implement (parts of) an application architecture and orchestrate components which are attached using so-called hot-spots [31]. Applications focus on providing complete solutions.

Moreover, functionality classification is refined by distinguishing application domains (that correspond to different market segments). Both components and frameworks can implement application-specific or generic features and thus belong to vertical (application) domains or horizontal (system) domains. Applications (by their nature) provide application-specific features. CompoNex provides a systematic structuring of component domains in the form of taxonomies (e.g. UNSPSC [35] to classify application-specific vertical domains).

In order to classify the component reuse model, the CompoNex component model additionally differs between logical and physical component reuse. Components, which are being offered for logical reuse, have to be delivered to the respective buyer (e.g. by shipment or download). Thereafter, the buyer installs the acquired component and assembles a local copy of the component during configuration. In order to use the component, the buyer usually has to pay a fixed price. On the contrary, physical reuse implies that component buyers remotely invoke component services, e.g. by using the XML Web service [6] or CORBA [25] technology. The respective seller usually hosts the component and accounts with buyers on a pay-per-use basis.

5 Related Work

The importance of facilitating component exchange between component sellers and buyers has been recognized as an important success factor of component-based software engineering for some time [31]. Subsequently, a variety of research and industry programs have been initiated to support the emergence of component markets [4].

As a result, prototypic component search engines, brokers, and marketplaces evolved in academia, e.g. the Agora Search Engine for Software Components [29], ComponentXchange [36], and the Software Component Broker [5]. Complementary, a variety of commercial component marketplaces have been developed and, until now, many of them have stopped their operations again [13]. A frequently updated overview that lists currently operational marketplaces is maintained at http://www.tu-chemnitz.de/wirtschaft/wi2/projects/components. However, neither academic nor industrial marketplaces offer the services that have been tied together as a result of the market analysis within this paper.

While the significance of component specification frameworks for the development of component marketplaces has been recognized in academia (e.g. [4], [7], [36]), none of the currently operational component marketplaces features a sophisticated framework to provide detailed information about components. This, however, makes it almost impossible for potential component buyers to assess components prior to acquisition (cf. chapter 2). In addition, none of the analyzed specification frameworks provided the information necessary to support the identified tasks (i.e. component identification, certification, acquisition, and assessment) completely. This has been

recognized by the standardization group that developed a more refined approach on component specification [1].

The newly developed reference marketplace CompoNex thus has a unique design that is based on a detailed market analysis and builds upon a sophisticated specification framework to provide information about components. Consequently, it implements a variety of improvements compared to existing marketplaces.

6 Conclusions and Future Directions

Despite lucrative forecasts for component markets and numerous research and industry programs, the expected (r)evolution still has to happen. To provide an explanation, today's component markets have been analyzed from a business perspective and a variety of immaturities have been identified. They have been used as a methodical basis to develop (some aspects of) a business model for component marketplaces by devising critical success factors to cope with identified market immaturities.

Moreover, these critical success factors have been refined and concrete proposals for implementing them as (modular) marketplace features have been provided. In order to evaluate the proposed improvements in practice and to better identify additional success factors in component markets, CompoNex is currently being designed as a model marketplace and, in an initial version, implements the here-mentioned features.

As component markets continue to evolve, the identified market characteristics are likely to change. Therefore, the provided business plan will have to be continually refined in future and new features will have to be identified.

Acknowledgements

The authors would like to thank Steffen Becker and Rolf Nebhuth for their hard work on the initial prototype and the members of the Oversoft Software Team for their valuable support during the design and implementation of CompoNex.

References

1. Ackermann, J., Brinkop, F., Conrad, S., Fettke, P., Frick, A., Glistau, E., Jaekel, H., Kotlar, O., Loos, P., Mrech, H., Ortner, E., Overhage, S., Raape, U., Sahm, S., Schmietendorf, A., Teschke, T., Turowski, K.: Standardized Specification of Business Components. German Society of Computer Science (2002) http://wi2.wiwi.uni-augsburg.de/fachkomponenten/memorandum.php?lang=eng
2. Akerlof, G. A.: The Market for "Lemons": Quality Uncertainty and the Market Mechanism. In: The Quarterly Journal of Economics 84 (1970): 488–500
3. Ansoff, H. I. (ed.): From Strategic Planning to Strategic Management. Wiley and Sons, New York (1976)

4. Aoyama, M.: CBSE in Japan and Asia. In: Councill, W. T., Heineman, G. T. (eds.): Component-Based Software Engineering: Putting the Pieces Together. Addison Wesley, Upper Saddle River, New Jersey (2001): 213–226
5. Aoyama, M., Yamashita, T.: A Commerce Broker of Software Components and Its Experience. In: Proceedings 2nd IEEE International Enterprise Distributed Object Computing Workshop (EDOC). IEEE Communications Society Press, Los Alamitos, California (1998): 155–161
6. Austin, D., Barbir, A., Garg, S. (eds.): Web Service Architecture Requirements. W3C Working Draft (2002) http://www.w3.org/TR/wsa-reqs
7. Beugnard, A., Jézéquel, J.-M., Plouzeau, N., Watkins, D.: Making Components Contract Aware. In: IEEE Computer 32 (1999) 7: 38–45
8. Blevins, D.: Overview of the Enterprise JavaBeans Component Model. In: Councill, W. T., Heineman, G. T. (eds.): Component-Based Software Engineering: Putting the Pieces Together. Addison Wesley, Upper Saddle River, New Jersey (2001): 589–606
9. Brown, A. W.: Large-Scale Component-Based Development. Prentice Hall, Upper Saddle River, New Jersey (2000)
10. Councill, W. T., Heineman, G. T. (eds.): Component-Based Software Engineering: Putting the Pieces Together. Addison Wesley, Upper Saddle River, New Jersey (2001)
11. Czarnecki, K., Eisenecker, U. W.: Generative Programming: Methods, Tools, and Applications. Addison Wesley, Upper Saddle River, New Jersey (2000)
12. Ewald, T.: Overview of COM+. In: Councill, W. T., Heineman, G. T. (eds.): Component-Based Software Engineering: Putting the Pieces Together. Addison Wesley, Upper Saddle River, New Jersey (2001): 573–588
13. Fettke, P., Loos, P., Viehweger, B.: Komponentenmarktplätze – Bestandsaufnahme und Typologie (in german). In: Turowski, K. (ed.): Tagungsband 5. Workshop Komponentenorientierte betriebliche Anwendungssysteme. Augsburg University Press (2003): 1–15
14. Flynt, J., Desai, M.: The Future of Software Components: Standards and Certification. In: Councill, W. T., Heineman, G. T. (eds.): Component-Based Software Engineering: Putting the Pieces Together. Addison Wesley, Upper Saddle River, New Jersey (2001): 693–708
15. Gamma, E., Helm, R., Johnson, R., Vlissides, J.: Design Patterns: Elements of Reusable Object-Oriented Software. Addison Wesley, Reading, Massachusetts (1995)
16. Garlan, D., Allen, R., Ockerbloom, J.: Architectural Mismatch: Why Reuse Is So Hard. In: IEEE Software 12 (1995) 6: 17–26
17. Gebauer, J.: Between Flexibility and Automation: An Evaluation of Web Technology from a Business Process Perspective. Center for IT & Marketplace Transformation, University of California at Berkeley (1999) http://www.ascusc.org/jcmc/vol5/issue2/gebauer.html
18. IEEE (ed.): STB610.12-1990 IEEE Standard Glossary of Software Engineering Terminology. IEEE Press (1991)
19. Jevons, W. S.: The Theory of Political Economy. Penguin Books, Harmondsworth (1970)
20. Katz, M. L., Shapiro, C.: Network Externalities, Competition, and Compatibility. In: American Economic Review 75 (1985): 424–440
21. Malone, T. W., Yates, J., Benjamin, R. I.: Electronic Markets and Electronic Hierarchies. In: Communications of the ACM 30 (1987) 6: 484–497
22. Meyer, B.: Object-Oriented Software Construction. Prentice Hall, Cambridge (1988)
23. Nelson, P.: Information and Consumer Behavior. In: The Journal of Political Economy 78 (1970) 2: 311–329
24. North, D. C.: Institutions, Institutional Change and Economic Performance. Cambridge University Press, Cambridge (1991)
25. OMG (ed.): CORBA Components. Overview and Specification: Version 3.0. OMG, Framingham. http://www.omg.org/cgi-bin/doc?formal/02-06-05.pdf
26. Overhage, S.: Towards a Standardized Specification Framework for Component Discovery and Configuration. In: Bosch, J., Szyperski, C., Weck, W. (eds.): Proceedings of the Eighth

International Workshop on Component-Oriented Programming (WCOP 2003). http://research.microsoft.com/~cszypers/events/wcop2003/09-overhage.pdf
27. Prieto-Diáz, R.: Implementing Faceted Classification for Software Reuse. In: Communications of the ACM 34 (1991) 5: 89–97
28. Sametinger, J.: Software Engineering with Reusable Components. Springer, Berlin (1997)
29. Seacord, R. C., Hissam, S. A., Wallnau, K. C.: Agora: A Search Engine for Software Components. Technical Report CMU/SEI-98-TR-011, Software Engineering Institue, Carnegie Mellon University (1998)
30. Shapiro, C.: Premiums for High Quality Products as Returns on Reputation. In: Quarterly Journal of Economics 98 (1983) 4: 659–680
31. Szyperski, C.: Component Software: Beyond Object-Oriented Programming. Addison Wesley, Harlow (1998)
32. UDDI Organization (ed.): UDDI Executive White Paper. UDDI Standards Organization Public Draft (2001) http://www.uddi.org/pubs/UDDI_Executive_White_Paper.pdf
33. UDDI Organization (ed.): UDDI Technical White Paper. UDDI Standards Organization Public Draft (2000) http://www.uddi.org/pubs/Iru_UDDI_Technical_White_Paper.pdf
34. UDDI Organization (ed.): UDDI Version 3.0. UDDI Open Draft Specification (2002) http://www.uddi.org/pubs/UDDI-V3.00-Open-Draft-20020703.pdf
35. United Nations Organization (ed.): The Universal Standard Products and Services Classification (UNSPSC). http://www.eccma.org/unspsc/crosswalk.htm
36. Varadarajan, S., Kumar, A., Deepak, G., Pankaj, J.: ComponentXChange: An E-Exchange for Software Components (2002). http://www.cse.iitk.ac.in/users/deepak/papers/ComponentXchange.pdf
37. Walras, L.: Elements of Pure Economics or the Theory of Social Wealth. Allen & Unwin, London (1954)
38. Weyuker, E. J.: The Trouble with Testing Components. In: Councill, W. T., Heineman, G. T. (eds.): Component-Based Software Engineering: Putting the Pieces Together. Addison Wesley, Upper Saddle River, New Jersey (2001): 499–512
39. Williamson, O. E.: Credible Commitment: Using Hostages to Support Exchange. In: American Economic Review 73 (1983) 4: 519–540
40. Williamson, O. E.: The Economic Institutions of Capitalism. The Free Press, New York (1985)