



CBPs are a kind of software componentry. These components can model, support and execute business process activities within an organization, such as problem-solving tasks. They can also interconnect and coordinate business processes across organizations, such as to support inter-organizational workflow. CBPs can be specified using high level scripting languages [O98] that are extended and tailored for modeling computer-based business processes. CBPs specify a control flow sequence of actions (or transactions) that different organizational actors perform using available application tools that consume their required inputs in order to produce the provided outputs. In addition, CBPs are internally represented as directed attributed graphs that can be externalized as semantic hypertext networks [GS89, NS91, MS96]. This enables navigational traversal that in turns provides a familiar mode of user interaction for browsing and enacting CBPs. Furthermore, using the capabilities of scripting languages to glue applications and tools together with their typed input and output allows CBPs to be interpreted (or executed) with computer support. Thus we find that essentially any computer-supported business can be specified and executed to some degree using CBPs.

CBPs link and integrate the products, people/roles, applications, heterogeneous information repositories and network computing environment through an *organizational process architecture* [SM97, NS97b]. An OPA serves as a conceptual and representational framework for specifying how individual CBPs can be configured and interconnected through their inputs, outputs, and other bindings [MS96, NS97a]. As a framework, it is possible to compose appropriate CBPs that provide computer-based support for common business processes such as purchasing, accounts payable, accounts receivable, and other corporate financial operations [SM97]. Such a framework of CBPs may then be reused and specialized in different organizational settings. Thus, OPAs provide a foundation for organizing CBPs as reusable software components for EC.

CPBs and OPAs are computational representations that can be interpreted. They also serve as prescriptive guidelines for how people in organizational settings perform their work. As such, we can characterize the kind of run-time environment that can support the development and execution of CBPs and OPAs. In simplest terms, these are called, *process-directed intranets* [SN97]. PDIs denote a network of application and repository servers, data transformation mediators, and end-user clients (Web browsers, integrated tools, and helper applications) that traverse a distributed semantic hypertext [NS91, NS97b]. PDIs in different organizational settings can be configured and interconnected to form *process-directed extranets* that realize *virtual enterprises* [NS97a, NS97b, SN97]. Thus, it is possible to rapidly prototype and engineer PDIs and VEs across their life cycle in a wide-area environment [SM97]. Similarly, PDIs and VEs can support the redesign and reengineering of business processes to which they are applied [SN97].

Finally, CBPs, OPAs, PDIs, and VEs are software technologies that are essentially neutral to the choice made for lower-level software interoperability mechanisms and data definition standards. Technologies and standards such as CORBA/DCOM, Java/ActiveX, EDI X12 transaction standards, UML and others are not necessary for CBPs. Alternatively, it is unclear whether these technologies and standards are necessary or sufficient for specifying or executing business processes in a coherent and tractable manner. However, these technologies and standards can be accommodated within CBP technology. In short, CBPs, OPAs, PDIs, and VEs can be developed and executed with or without the software technologies and standards noted above. Thus, it seems reasonable to posit and discuss how CBPs can serve as a foundational software technology for EC along side of the other technologies and standards that have been proposed so far.

References

[GS89] P.K. Garg and W. Scacchi. ISHYS: Designing Intelligent Software Hypertext Systems. *IEEE Expert*,

4(3):52-63, 1989.

[MS96] P. Mi and W. Scacchi. A Meta-Model for Formulating Knowledge-Based Models of Software Development. *Decision Support Systems*, 17(3):313-330, 1996.

http://www.usc.edu/dept/ATRIUM/Papers/Process_Meta_Model.ps

[NS91] J. Noll and W. Scacchi. Integrating Heterogeneous Information Repositories: A Distributed Hypertext Approach, *Computer*, 24(12):38-45, December 1991.

http://www.usc.edu/dept/ATRIUM/Papers/Distributed_Hypertext.ps

[NS97a] J. Noll and W. Scacchi. Supporting Distributed Configuration Management in Virtual Enterprises. *Software Configuration Management*, R. Conradi (ed.), Lecture Notes in Computer Science, Volume 1235, Springer-Verlag, New York. 142-160, 1997. <http://www.usc.edu/dept/ATRIUM/Papers/DHT-SCM7.ps>

[NS97b] J. Noll and W. Scacchi. Supporting Software Development Projects in Virtual Enterprises. *Journal of Digital Information*, (revised version, to appear). <http://www.usc.edu/dept/ATRIUM/Papers/DHT-VE97.html>

[O98] J. Ousterhout. Scripting: Higher-Level Programming for the 21st Century. *Computer*, 31(3):23-30, March 1998.

[SM97] W. Scacchi and P. Mi. Process Life Cycle Engineering: A Knowledge-Based Approach and Environment. *Intern. J. Intelligent Systems in Accounting, Finance, and Management*, 6(1):83-107, 1997.

http://www.usc.edu/dept/ATRIUM/Papers/Process_Life_Cycle.html

[SN97] W. Scacchi and J. Noll, Process-Driven Intranets: Life-Cycle Support for Process Reengineering, *IEEE Internet Computing*, 1(5):42-49, September 1997. <http://www.usc.edu/dept/ATRIUM/Papers/PDI.ps>