Imagine that an intelligent, friendly extra-terrestrial creature has just landed on planet Earth, without any prior knowledge of earthly physical and societal systems. Trying to figure out what his senses are telling him and how to construct a model of the surrounding reality, this creature is in a situation not unlike that of a human system developer, who is beginning to evolve a new system in an unfamiliar domain. Lacking sufficient knowledge about the domain that hosts the system and its environment, the analyst must start with an arbitrary collection of facts. The collected observations are not overly refined nor are they extremely abstract. The knowledge and understanding of the system is gradually improved through activities such as observing the current state of affairs and practices, inquiring, interviewing professionals in the field, and reading relevant documents.

Soon enough, details about the system begin to mount. An intuitive yet formal way of documenting this growing amount of collected information must be in place. Subsequent, higher order cognitive activities follow, including understanding, modeling, analyzing, designing, presenting and communicating the analysis findings and design ideas. These mental activities rely heavily on a solid infrastructure for organizing the accumulated knowledge and creative ideas. Long before the system becomes particularly intricate, it is very helpful to display these results in a medium other than the brain.

Systems and products are becoming increasingly complicated. Technology has been so pervasive that even commonly used products feature high computational power, embedded within increasingly miniature, precise and involved hardware. Systems of an infrastructure nature, such as air traffic control, the Internet, and electronic economy, are orders of magnitude more complex than products individuals normally use. Understanding natural, artificial, and social systems requires a well-founded, yet intuitive methodology that is capable of modeling these complexities in a coherent, straightforward manner. The same methodology should be useful for designing new systems and improving existing ones. These systems are intricate enough as it is; there is no need to add confusion by using a method that is itself complex.

Artificial systems require development and support efforts throughout their entire lifecycle. Systematic specification, analysis, design and implementation of new systems and products are becoming ever more challenging and demanding, as contradicting requirements of shorter time-to-market, rising quality, and lower
cost, are on the rise. These trends call for a comprehensive methodology, capable
of tackling the mounting challenges that the evolution of new systems poses. The
development of Object-Process Methodology was motivated by this call.

The idea of developing systems in a unified frame of reference is not new. For
example, Bauer and Wössner (1981) noted that systematic development of basic
concepts leads to methods that cover the entire system’s lifecycle. Object-Process
Methodology, or OPM for short, takes a fresh look at modeling complex systems
that comprise humans, physical objects and information. OPM is a formal para­
digm to systems development, lifecycle support, and evolution. It caters to people’s
intuition and train of thought. Preliminary ideas of OPM were expressed in (Dori,
1995; 1996) and applied in such diverse areas as computer integrated manufactur­
ing (Dori, 1996A), image understanding (Dori, 1996B), modeling research and
development environments (Meyersdorf and Dori, 1997), algorithm specification
(Wenyin and Dori, 1999), document analysis and recognition (Dori, 1995A;
Wenyin and Dori, 1998), and modeling electronic commerce transactions (Dori,
2001).

Natural and artificial systems alike exhibit three major aspects: function (what
these systems do), structure (how they are constructed), and behavior (how they
change over time). Since OPM does not make any assumptions regarding the
nature of the system in question, it can be applied in any domain of human study or
endeavor. In the case of artificial systems, it provides a framework for the entire
system’s lifecycle, from the early stages of requirement elicitation and analysis,
through further development and deployment, all the way to termination and initia­
tion of a new generation.

OPM combines formal yet simple graphics with natural language sentences to
express the function, structure, and behavior of systems in an integrated, single
model. The two description modes OPM uses are semantically equivalent, yet
appeal to two different parts of the brain, the visual and the lingual. OPM is a prime
vehicle for carrying out the tasks that are involved in system development. It does
so in a straightforward, friendly, unambiguous manner. The design of OPM has not
been influenced by what current programming languages can or cannot do, but
rather, what makes the most sense. Due to the resulting intuitiveness, OPM is com­
municable to peers, customers and implementers. At the same time, the formality
of OPM makes it amenable to computer manipulation for automatically generating
large portions of the conceived system, notably program code and database
schema.

Where does the name Object-Process Methodology come from? Objects and
processes are the two main building blocks that OPM requires to construct models.
A third OPM entity is state, which is a situation at which an object can be and
therefore a notch below object. Objects, processes and states are the only bricks
involved in building systems. The links connecting these three entities act as the
mortar that holds them together.

As for methodology, Computer Desktop Encyclopedia (2001) defined it as “the
specific way of performing an operation that implies precise deliverables at the end
of each stage.” OPM specifies a way of understanding and developing systems. Its