IT Modeling Experience in Urban and Regional Development

Marija MARUNA & Vladimir MARUNA

Marija Maruna, MSc Arch, Belgrade University, Faculty Of Architecture , Bul. Kralja Aleksandra 173, Belgrade, m.ma@sezampro.yu,
Vladimir Maruna , BSC EE, Dynamis, Molerova 70/I, Belgrade, vmaruna@eunet.yu

1 INTRO

Contemporary world increased dynamics forcing accordingly every human activity leaving none outside this never-ending race. Regional development is certainly strongly influenced by this trend challenging regional and urban planning techniques to cope with that. Acceleration of a working context and ambient accompanying with significant complexity are reasons, not only reasons however, for new evaluation and consequent advancement of current urban development planning methodologies. Shortage of a new appropriate techniques and methodology is apparent. Methodology that is able to respond quickly and efficiently to changes, independently from social and cultural context, toward to current development trends.

Current planning techniques are based on accustomed mixture of verbal and informal graphic and symbolic language. In the same time research substance if exceptionally complex, heterogeneous and substantial retaining numerous different parameters with rich relationships matrix, required to consider and study in order to fulfill the designated outcome through firmly defined set of activities. One of most difficult problems is spatial dynamics since it is hard to recognize and identify and accordingly difficult to describe and control.

Consequently, advancement of a planning methodology is essential to manage all listed inconsistence and weakness i.e. a new planning methodology is mandatory. This objective will certainly require a fundamental breach and change of current planning techniques demand and new interdisciplinary approach and research.

Similar complexity level, vast list of parameters with complex and heterogeneous structural and dynamic relationship scheme, may be found in information and telecommunication systems i.e. computer based systems. Information and Telecommunication Technology (ICT in further text) uses object oriented methodology, for analysis and design as well, (OO in further text) to cope with the complexity, scope and heterogeneity of a systems. Latest step in this direction is a Unified Modeling Language (UML in further text) as a cohesive graphic and formal language for system description. UML is adopt as an industry standard and today is almost inevitable in every even modest ICT project.

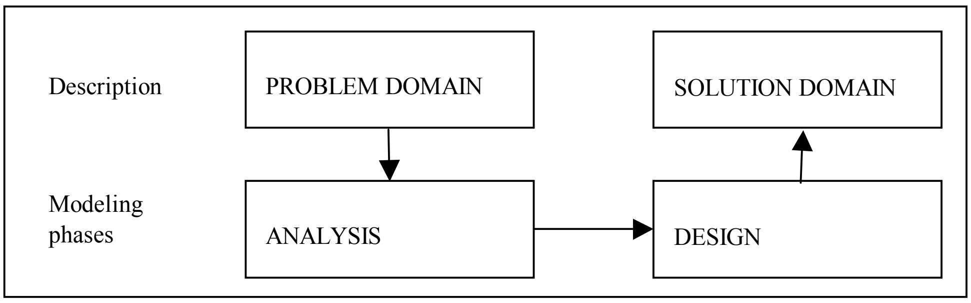
Paper illustrates basic building block of OO methodology and UML, and theirs many years of positive experience and encouraging results, intending to exploit this in an urban spatial planning domain. Higher perception and description level, enhanced knowledge and improved control over the urban development are main objective. However authors do not consider direct, as is, implementation of OO methodology and UML to the problem of urban and regional development. Localization and adjustment to the problem domain is advisory. This is already accomplished for several other vertical industries as joint effort of local experts and OO/UML professionals.

Main goal is to apply currently acquired OO/UML experience and knowledge to the urban and regional development problem domain. This interdisciplinary research will be, evidently, long, complex and expensive process but we believe that this direction promises a significant progress for urban planning. Urban ambient formal specification, founded on OO/UML modeling methodology, may accumulate experience and knowledge presenting it to the audience, introducing computer added urban planning and, may be in the future, urban ambient simulation.

2 MODELING

Firm understanding and knowledge about certain phenomenon is essential precondition for its control i.e. control of its development. ICT modeling is a good methodology for analysis i.e. comprehension and description of complex system structure and dynamics and this modeling could facilitate the same for spatial phenomenon's, conceived as complex systems, like city, region and others.

Modeling for analysis and design of a complex system is inevitable first step in modern engineering. Analysis is process of mapping of problem domain to the model i.e. its formal representation using techniques like abstraction, conceptualization, simplification, decomposition and similar. Design is process of mapping of analytic system representation, result of analysis, into the implementation model i.e. translation from problem to the solution domain. Implementation model is afterward used for real system component construction and consequently for system existing in a real world. However, either in case of analysis or design, modeling will result into a set of models, which are formal, simplified and conceptual representation of reality. This set of models is dividing in two subset named analysis model and design model (G1)



Graph 1: Modeling phases

2.1 Modeling goals

Previously listed models, analysis and design models, are presenting structural, dynamic and other characteristics of targeted system. Analysis model will facilitate problem domain understanding and design model proposes the solution to implement to solve the problem. Anyhow, model describes the systems independently of its purpose. Accordingly, system modeling will have following objectives and goals:

- Description of system working context i.e. domain of system existence,
- Description of internal system dynamics and system dynamic interactions with working context,
- Description of internal system structure and system structural interactions with working context,
- Description and specification of system components and theirs structural and dynamics interconnection which is in ICT modeling named system architecture,
- Refined comprehension and detailed overview over the system as a whole and partially through its components from different aspects i.e. separate aspect for every previously identified and defined system actor,
- Formal, precise, complete and efficient communication among analysis and/or design modeling team members and
- Asynchronous, parallel and coordinated cooperation, through certain predefined activities, between analysis and design modeling team members.

Models are described with formal and consistent language, which is independent from domain, used verbal language and particular industry context. This establishes precise and efficient team member communication even between experts from different domains. This is of essential importance for overcoming over cultural and experience diversity, which are permanent source for misunderstanding especially in researches of complex phenomenons with heterogeneous and interdisciplinary teams.

Models are simplification of reality since they leave all unnecessary details out. This, of course, leaves possibility to discard valuable details of reality, may be not essential but important for a better solution and/or comprehension, resulting in incomplete model perception of a reality. Derived conclusions and designed solution will then have lesser chances for success. On the other side model may include elements, which are not required, elements that not contributing to the quality of proposed solution, making model redundant and additionally complex to understand and manage. Good model will include all valuable attributes of representing reality discarding all redundant for a given level of abstraction.

2.2 Benefits of Modeling

Modeling benefits could be derived from previous list i.e. list of objectives and goals however; we will list it once more because of its importance:

- **Abstraction and Simplification** leaving redundant and insignificant details out of a scope, classified as such by means of end modeling objective i.e. purpose of a system, enabling unambiguous focus on system important aspects,
- **Decomposition** unscrambling wide and complex system to a smaller and less complex subsystems, with firm and defined interface between them, allowing full attention to this simpler and easier to cope with parts which is basement for solving of complex problem and it is also natural approach for human intelligence,
- **Communication** using model as formal language independent from local context and experience, empowering firm vertical and horizontal team integration which is almost only successful approach for acceptable level of understanding for interdisciplinary teams,
- **Cooperation** since model defines and facilitates separation of responsibilities enabling proper coordination of conducted activities, **iteration** using model as a starting point for every subsequent step in problem solving and **accumulation** because model contains all previous iteration, therefore all collected knowledge and experience, that is all previous work on problem solving,
- **Unification and standardization** in view of the fact that is model specified with unified language, which does not have to be UML, imposing same principle i.e. use of standards and formal descriptions on the whole process level and to all team members and
- **Documenting** because model holds all significant facts about the system, working context, problem domain and problem solution serving to the every and all model users at any time, accordingly to hers/his role, required level of details and wanted aspect, as comprehensive knowledge base.

2.3 Main Modeling principles

Modeling, as engineering technique, have productive and long history and experience resulting in several basic principles positively evaluated through it. Although these principles are emerged from ICT problem solving process, we consider them easy applicable to any vertical industry because they are immanent to the modeling not to the ICT.

First is a **choice what to model** which have great influence to the solution shaping and designing. Consequently, right model will precisely and brightly reveal all essential aspects of a problem resulting in a quick and sub-optimal solution. Partially or fully incorrect model choice will postpone or even seriously obstruct solution of a problem. Consequently, right choice what to model is essential and has to be seriously considered at the beginning of system modeling.

Secondly, is that every model **specifies selected aspect** of a system **on appropriate level** of details. Ambition is to specify every system aspect on the level that is suitable for particular user i.e. model will contain only those elements that are interesting for particular user. Excellent model will offer all and any of its users to choose appropriate level of details accordingly to his/hers current team responsibilities

Model is always **simplification of reality** but has to **stay connected** to that reality meaning that model will have to specify what is simplified from reality in its specification together with the level of simplification,. This will enable constant check of ignored details re-evaluating theirs importance and value for defined system purpose. This will allow extension of a model, during its development, with previously ignored details if theirs re-evaluation classify them as valuable, improving accordingly, the quality of a system description and designed solution.

Complex and heterogeneous system are best described with **small set of independent or lightly dependent models** because it is hard and useless task to try to represent them with only one model. It is extremely confusing and hard to cope with numerous aspects, entities, relationships that are immanent to complex systems. Set of models are better, since every single one may try to highlight a particular aspect or system functionality. Those models are then connected through meta-model (See description below). Certain system will give preferentiality to certain models while others will use other approach depending on nature and scope of a modeling system.

3 OBJECT-ORIENTED MODELING

ICT modeling techniques and praxis recognizes two major type groups, of methodologies, based on theirs baseline approach i.e. algorithmic and object oriented. Object oriented methodologies are prevailing because theirs easy adoption to almost any problem domain and level of complexity. Moreover, transition from analysis to the design model is significantly easy and forthcoming overcoming strong boundaries between them. This is probably main reason for theirs attractiveness.

Basic entity of an OO modeling is an **object** and **class**. Object is real tangible thing existing in a reality and class as the formal specification and abstraction of an object set. Class will specify all common attributes, methods, relationships and semantics of selected object set. System domain is, in this first step, separated to the valuable objects, theirs structural and dynamic relationships and semantics for selected system behavior. This defines vocabulary of a system domain. Second step is abstraction and conceptualization of observed objects through theirs classification. Recognized classes will then establish relationships correspondingly to the relations between objects. These relations, in OO modeling theory, could be one of the following: classifications, generalizations/specializations, associations or aggregations/compositions.

Classification is a process of identification of a set of objects with common attributes, dynamics and semantic within the system domain. This identification strongly depends on a purpose of a system and it is, therefore, always domain specific. Results of a classification are classes. Object is than an **instance** of a certain **class**. Classification will establish relations between two sets i.e. a set of objects and a set of classes, defining for every object and every class if selected object belongs to the selected class. OO methodology does not recognize an object without belonging class although class may have none object instances.

Next level of model development is establishment of other relations between previously recognized classes and usually first on the list if building of the **class hierarchy**. OO modeling uses two complementary methods to establish class hierarchy named generalization and specialization. Generalization defines more abstract concepts i.e. deepens system conceptualization by abstracting the differences and forcing the commonality over particular set of selected classes. Result of a generalization is a new class, more general than starting set, comprising all common elements of started set. Resulting class is called **ancestor** or **parent** while all classes from starting set are **descendant** or **child**. Opposite method will extend selected class with more specialized attributes and/or semantics resulting in class that is more specialized. Resulting class inherits all attributes, methods and semantics from starting class adding ever more to it. Resulting class is descendant or child while starting class is named ancestor or parent. Those two methods will form a hierarchy tree where every descendant inherits everything from its ancestor. This configuration of a system domain facilitates efficient and precise conceptualization of it and it is standard practice in OO modeling. Classes from the top of this tree, classes without ancestors, are called power classes or **power types**. They are usually parts of a meta-model (see below about meta-model).

Complex system structure could not be defined and specified with only generalization/specialization relations. These relations are not sufficient to represent all aspects of class structural relations. **Aggregation**, as additional and important relation, is relation between part and a whole. Aggregation will arrange a new class from a set of classes. New class is a whole while used classes are parts. Aggregation reduces class complexity focusing to a whole and abstracting the details and complexity of its parts. Many of attributes and methods applicable to a whole are applicable to the parts too. Those who are not are **propagated** from a whole to every part for local interpretation. Aggregation is used wherever is useful to address a whole, instead every part, decreasing the complexity of a system representation. Description of a complex system will almost certainly include special kind of an aggregation named **composition**. Composition defines a whole as immutable configuration of parts i.e. configuration that could not be changed during object/class life cycle. Composition is often considered as aggregation that “contains” parts while usual aggregation is aggregation that “references” parts. OO modeling theory recognizes six aggregation types. Further info about the aggregation and attributes/methods propagation may be found in L2.

OO modeling recognizes relations, which are not listed above, but still significant and valuable for system understanding and control. Those relations are named simply **association** representing any structural and behavioral dependency between classes. One may define aggregations as a special kind of association since association set includes aggregation set. Moreover, class attributes are actually association of a class with other classes in a system domain.

Every, previously listed, relation will follow a set of constraints that are applied to them. OO modeling recognizes a wide set of applicable constraints but we will here define only a few. Reader may find others in L2, L3, and L8. First significant constraint is cardinality defining exact number of class instances allowed in a relation, which is defined by a range starting from none ending with many. Next important set o constraints are theirs mathematical characteristics that is relation may be reflexive, symmetric, asymmetric, transitive etc.

All previous defined building blocks will define system structure, although certain dynamic aspects may implicitly be included. Complete and precise system specification requires its behavior specification consequently additional elements for system dynamic description are required.

System behavior and dynamics may be precisely defined with set of states and transitions between them. System is, on the other side, composed from objects therefore system state is represented with state of every containing object. Object state is represented with current value of its attributes. If we recall previous definition of an object attributes as a set of association we will define an **object state** as current **set of its associations** with other objects. Association set will also classify an object to the class therefore **object state** is represented also with a **set of classes** to which is object classified. Object **state transition** is actually association/class set change. It is usually crucial, for complete and precise comprehension of a system, to specify all states and transitions that object may have within the system. That is object's **life cycle**. Model will contain life cycles only for important objects and set of objects. Same approach may be used for object, set of objects, subsystem or system in it entirety increasing the abstraction level accordingly. Life cycle is specified with **finite state machine** containing definition for both, states and transitions. System behavior may be additionally described with **objects collaborations** within particular system function and/or response or with resulting **sequence of events**. Object transition is result of a stimulus from outside of a system or inside i.e. result of a change of other object state. Object state is changed with object method. OO modeling recognizes methods as the only way for object state transition.

System specification is not complete with only structural and behavioral representations i.e. without specification of **domain specific rules**. Rules are specifying a particular politic conducted within the system during its functioning as well as its mandatory requirements. Every rule has to be valid at every time during system life cycle and under any circumstances. Rules may be defined as constraints and derived rules. **Constraints** are rules that establish boundaries over a set of selected objects that is over theirs behavior, structure, relationships etc. **Derived rules** are statements of type "if then" or "only if then" or formulas describing how to calculate or derive certain values or states. Although is this usual and standard classification of domain specific rules it is not the only one. Anyhow, rules should be defined using formal and defined language toward theirs computer processing, representation or application.

Before we switch to the UML, as one OO modeling language, we will describe three abstraction levels which are immanent in OO modeling and which are previously implicitly referenced. Those are **meta-model**, **model** and **object levels** of abstraction. Meta-model level includes definition of all object types and elements used on a model level. Meta-model is a model of a model. Entities like class, object, relation, attribute, generalization, specialization, association, aggregation, composition, basic data types etc are all fully defined on a meta-model level. Meta-model level is entirely abstract, representing the OO modeling methodology itself. Second level is a model level, which is previously presented. Model is an instance of a meta-model and model classes are instance of meta-model classes. Last level is object level, which was starting point for whole abstraction process. Object model, if exists, is an instance of a model and objects are instances of classes. Presented hierarchy is standard for contemporary CASE tools.

4 ELEMENTS OF UML

UML, Unified modeling language, is standardized object-oriented language for visualization, specification, construction and documenting or artifacts of computer based systems.

UML is a language for **visualization** of system entities and behavior resulting in graphic representation of concepts. This is baseline for communication between team members even with different domain origin and for enhanced and precise understanding of treated problem.

UML is a language for **specification** i.e. for precise, formal and complete definition and description of system elements and relations between them as well as rules which are valid on a system level.

UML is a language for **construction** because it is highly formal and it could be easily coupled with particular development environment as well as with specific domain techniques and activities conducted in the real world

UML is a language for **documenting** used for description of all system elements including its requirements, constraints, domain specific rules, system design, project plan, roles, responsibilities, system architecture and similar. All analysis/design artifacts are documented with UML and included into the model where every user may approach to them on a way and levels of details applicable to its role.

UML is a language and is therefore defined with vocabulary and, accompanying, semantic and syntax. UML Model is well formed if it conforms to proposed syntax. UML model is formal statement or set of statements about the system used for communication purposes. Modeling language is language with vocabulary and grammar adjusted to the description of modeling system conceptual and physical representation. Modeling objectives are reality abstraction, simplification and conceptualization as first step toward system comprehension. UML defines how to form well-formed model and how to read it. UML does not define what, when and how to model. It is defined in a modeling methodology, which is based on UML like for example Rational Unified Process, Catalyst or others.

UML is highly formal, extendible language, where extension will not decrease its formality, and also highly intuitive and therefore easy to learn and manage at least for reading. UML is defined with its conceptual model with three groups of major elements named: **basic building blocks** (nouns, verbs and semantics), **composition rules** for building blocks (language grammar) and **common mechanisms** that apply throughout the UML. UML vocabulary includes three kinds of building blocks called: **things**, **relationships** and **diagrams**. Things are basic object-oriented building blocks of UML including objects, classes, interfaces etc. Generally, things will include structural, behavioral, grouping and notational types. Relationships will include all previously described and few additional like realization. Diagrams are graphical representation of a selected set of elements used for system visualization from particular perspective establishing a projection into the system. Further details about the UML conceptual model and its elements may be found on L2.

UML's formality, simplicity and extensiveness is powerful driver, recognized by authors of this papers, for a shift of an UML scope from a domain of computer based systems to other specific problem domains including urban and regional planning as well. This will transfer all positive OO modeling experience to other complex domains of human interests and probably will improve level of knowledge accumulation and transfer, controllability, etc. Authors **do not** consider UML, as state of the art, to be directly applicable to urban and regional planning but rather suggest its adaptation to this specific domain. Definition of a regional and urban meta-model may be good first step.

5 PROSPECTIVE APPLICATION OF OO MODELING IN URBAN AND REGIONAL DEVELOPMENT

Main objective of this paper is to promote OO modeling methodology and accompanying positive experience for analysis and design of a domain of urban and regional development i.e. its modeling. OO Modeling will probably start with the analysis of urban space domain and its concepts identifying and specifying its basic entities and theirs relationships. This may end with valuable and practical taxonomy of an urban space. This taxonomy will be almost certainly incomplete and unfinished but it may be firm starting point for establishment of a corresponding UML meta-model. Current planning methodologies already implicitly use models and modeling approach but models and those techniques are not clearly stated and definitely are not unified. Moreover, planning process is already full of conceptualization and abstraction activities but they stay wasted in planners mind entering the real world only through informal and non-unified representations full of inconsistencies and localism. What we suggest here is to write down this mental modeling process directly to the customized UML models, customized to the problem domain, and to enable to the others direct approach to abstract concepts not to the vague presentation. This will not only overcome inevitable misunderstanding but also expose ideas to the public evaluation resulting almost certain in some improvement and enhancement. Moreover, models will continue to grow accumulating experience and knowledge and form exceptional knowledge base for coming generations.

Complex system requires formalized approach for powerful conceptualization and abstraction i.e. they require modeling approach to cope with complexity and volume. Modeling approach decomposes system to subsystems and then subsystems to components and so forth, concerning the interfaces between parts through this whole process. This will move analysis focus from whole system, difficult to manage, to less complex and manageable parts. Narrow focus leaves all unnecessary details out of scope enabling good conceptualization and abstraction. Modeling approach, opposite to other methodologies and techniques, emphasize relations between identified concepts and entities. On the other side urban and regional development is extremely complex and heterogeneous system with numerous and strong relations and mutual dependencies i.e. with numerous and complex relationships of different kind. Modeling approach with rich and precise vocabulary for all kinds of relations is therefore good starting point. Models, unlike GIS, will cover all functional relationships between entities, will describe all planning rules and suggestions, will describe global and local working context and therefore will represent full complexity of urban space from local and general point of view. Description of dynamic and behavior of a system, urban spatial structure, will facilitate anticipation of a development process and its phases ending with better control over it. Modeling of dynamics and behavior of urban spatial structure imply prosper establishment and use of dynamic regional and urban plans. This could facilitate better control and fine-tuning of development process based on acquired current state, defined planned state, by calculating the difference and offering corrective activities.

Applied OO/UML analysis will:

- identify basic concepts and entities of problem domain,
- establish classes and classification tree,
- identify and specify all valuable relationships between defined classes,
- identify and define system dynamics, states and transitions

In other word will establish complete system specification. Collected knowledge and generalizations will be later shifted on meta-model level as idiosyncratic knowledge about urban and regional development. This establishes particular abstract knowledge catalog used as design patterns.

Modeling will use proposed design patterns to model real world problem but will still stay on abstract level. Use of predefined, well-formed and described concepts will introduce at least less errors and mistakes in planning activities founded on accumulated knowledge and experience.

Last step is extraction of plans as instantiation of previously developed models targeting real world ambient and problem. Plans are like objects instantiated for particular real world and tangible things. Plans are customized re-description actual actors and actual ambient ready to use in a real world.

Meta-models, as it is explained above, are planners' knowledge base comprising all experience, knowledge, well techniques, even legislative rules specified as meta models entities. For proper definition of a meta-model, we will suggest iterative and incremental approach where every increment should include two steps:

- Extensive implementation and use of a meta-model through planning practice and
- Re-evaluation and enrichment of the meta-model based on a new acquired experience and evident important changes in the environment.

Meta-model is therefore never complete and it is always open to comply with the real world changes and requirements i.e. representing real world as much as it is required and applicable. Resulting meta-model should be considered as shared property and value of a whole industry and should be treated accordingly. On the same time, meta model must stay available and open to the audience treated as open source project, which is proven technique in modern world.

6 LITERATURE

Alexander, C., Ishikawa, S., Silverstein, M., A Pattern Language, New York: Oxford University Press
Booch, Grady, Rumbaugh, James, Jacobson, Ivar, The Unified Modeling Language User Guide, Addison-Wesley
Cheesman, John , Daniels, John , UML Components, A Simple Process for specifying Component-Based Software, Addison Wesley, 2001
Hillier, Bill. The Common Language of Space. www.spacesyntax/commonlang.html, 30.05.01.
Lane, Michael – Editor, Introduction to Structuralism. New York: Harper Torchbooks, 1970
Maruna, Marija. Istraživanje mogucnosti strukturalne metode u otkrivanju i definisanju prostornih obrazaca grada., Beograd, Arhitektonski fakultet (magistarski rad), 2003
Mc Loughlin, J.Brian , Urban and Rregional Planning A System Approach. London: Faber and Faber , 1969..
Odell, James J., Advanced Object-Oriented Analysis and Design using UML,Cambridge University Press, Sigs Books,
Urhahn, G.B., Bobic, M., A Pattern Image. Bussum: THOTH Publishers, 1994
Wilson, A.G., Rees, P.H., Leigh, C.M., Models of Cities and Regions. Chichester - New York - Brisbane - Toronto: John Wiley&Sons, 1977
Luhman, Niklas, Teorija Sistema, Plato, Boegrad, 1998