World-Wide Mind and eThings: Cooperative Virtual World

Augustin Mrazik
ARS NOVA s. r. o.
Brigadnicka 27, SK-84110 Bratislava, Slovakia
augustin.mrazik@arsnova.sk - www.arsnova.sk
fax: +421-2-60-300-212

Abstract

Internet has made a revolution that is often being compared to Guttenberg's invention of book print. However, from the point of contents, today's Internet still does not enable to communicate much more than print - words and pictures (albeit dynamic and generated on-the-fly from a database). However, people think in concepts and laws when reasoning about real world things and systems. We propose how these concepts as well as things could be represented on Internet in a live and working form, developed, shared and used for practical purposes by all Internet users. This approach would lead to a cooperative development of an environment containing the Common Sense (similar to Wikipedia, but in a live computational form) and to the seamless interconnection between the real world things and their virtual counterparties - e-things on Internet.

Keywords: ubiquitous computing, object model, knowledge representation, ontology, world-wide mind, spacetime, events, constraints, rules, space, topology

1. Vision of a Ubiquitous Virtual World

Already for a long time, I have a dream of "*World-Wide Mind*" (WWM) - a next generation Internet application based on a sole and simple, yet powerful concept - just like Smalltalk was a uniform computing environment based on a single and simple concept of objects, and it was still powerful enough to be useful for any new problem or application by extending the existing environment by the required delta. By teaching the live running system new things, new behavior in the form of ontology of the specific domain (category of classes), which was extending the shared upper ontology of Smalltalk as a computing environment. By describing the problem in an understandable way as communicating objects similar to the particular things from the problem domain, not by forcing to transform the problem to a form understandable by the computer.

Internet has almost become a virtual world. Today "everything" is on Internet. You can find here any kind of information, book a flight or a hotel, sell your stock, see a movie or post your movie sketch, talk to and see your spouse who is on the other side of the earth or participate in a virtual conference including a shared white board. Very many people had learned to use these applications. But don't try to look behind the screen, don't try to understand how it works or even to change it! In my opinion, you look at a nice facade of a movie decoration – the construction behind the facade is eclectic and it has almost no basement. It looks nice and actors move in and out, you can do it, as well. But it is just a question of time when new and new extensions of this poor building break down the whole thing.

Computers, programming languages as well as Internet were created in order to be used by people for practical purposed, to enable their communication and satisfy their need for information, to provide information about the real world, people, their problems and activities. But today's approach for such a model is based on technologies, which are suitable rather for computation than for simulation and modeling of things of such a complexity. Programming languages, databases and GUI technologies are "computation-centric" – they pay attention much more to what the computer can do than to what should be done for the users. But our aim is to model the real world, not to feed the computers!

Although we have Java, CSS, XML, AJAX, EJB and lots of other technologies nowadays, languages and systems for building Internet applications, the final result for the user are still "HTML" pages – albeit they are generated on the fly and filled with JavaScript and Flash. Internet became a new Tower of Babel – still new and new languages and technologies are introduced in order to fill some gap or to solve some problem, which suddenly arose, and they are making the basic problem still worse and worse. There are many voices warning before the collapse of computing environments.

My vision is a live environment on Internet, which will be built upon a simple, uniform and understandable concept - *eThing*. The "object model" of eThings is based on fine-grained separation of concerns, compatible with the way how people think and reason about the real world and its dynamics. Each substantial concern from the real world should have its counterparty in the "object model" of eThings. As an example – most real world things are spatial, they have their borders, which define their bodies and divide them from other things in their environment, or at least they have a place (location) or spatial coverage. Even concepts (abstract things) have some coverage where they are defined or valid – most general ones are simply valid in the whole space, i.e. everywhere. Further, people think (and act) rather in laws and rules than in algorithms and methods (albeit they know or have to learn many stereotypical procedures to have the ability to perform certain activities). Hence, if we want to model such common things as accidents, causality of actions, missed opportunities and that like, the behavior of eThings and dynamics of the whole system should include also laws, rules, constraints and events (beside methods and actions).

eThings will be live, active, they will be able to react to events in an "intelligent" way defined by rules and constraints. eThings will be the only construct for modeling anything and everything - from very abstract concepts to very concrete things, including numbers, date and time, money (currency) etc. Even "really static" things as documents or pictures will be represented by eThings – still they can define some rules or actions by which they can react to their unauthorized use or may delete themselves when not being used for a long time.

With a certain level of abstraction we can say, that WWM will be based on eThings just like WWW is based on HTML (incl. XHTML, CSS, XML etc.). But in contrary to WWW, business logic will be described by eThings and included in WWM directly, not in separate applications operating "on-top of the data". Contents of WWM will be created and maintained by the users themselves in a cooperative way similar to Wikipedia, with immediate feedback to problems and errors. Of course, kernel eThings of new application domains will be created by skilled domain experts (who will not necessarily be programmers), but standard users will be able to explore and understand the way how the things (eThings) work, they will be able to learn, grow and expand their knowledge with each use and later they may be able to correct or extend the contents in order to match the reality or their needs in a better way. For professional use, experts worldwide will be able to exchange and share the results of their work on-line in a form of a working model developed cooperatively across the whole world, test it on data provided in the same form by other professionals (e. g. as data from deployed databases) and publish final result in the form of a working knowledge base (which really works and can be used immediately by everybody), not only in the form of a paper (which can be understood only by experts of the same domain). This will fundamentally change the way how software and database applications will be developed and deployed. Even more - this will fundamentally change the way how knowledge is being developed, published, learned and reused.

WWM will enable much higher level of cooperation between people than known today. Internet was created as a means for communication between people and WWW has completed this aim. However, the contents of this communication is still provided almost in the same form as any new or old book - whether electronic, printed, or even hand-written and painted in the times before Guttenberg: the contents has to be encoded to words (and pictures), published, read, understood and correctly interpreted by the reader. I believe that eThing technology can change the communication between people to cooperation and co-thinking – creating one shared World-Wide Mind as a definitely virtual world, interconnected with the things of the real world and with people directly.

2. Introduction to eThings

Our aim, based on the vision, is to develop a knowledge management and computational environment, which will run on Internet and which can be shared and used by all people. (Under knowledge we understand also all kinds of information, data etc.) Such an environment will create a new quality for cooperation between people – sharing and jointly enhancing the knowledge of the humankind, which would be available on Internet on-line in a "native" interoperable form.

So far, the knowledge can exist solely in two forms: as text in books or in the "native" form in the human mind. The textual form of the knowledge (including picture, multimedia etc. in order to better describe some more complex things and relations) can be viewed as knowledge "encoded" or "hibernated" in words (and pictures), which have to be read and understood by a man in order to be converted to the native form. To understand the "isolated words and pictures" one has to have the necessary knowledge background in order to interpret in the right way the semantics, which is "hidden behind the words". This process of understanding the text is actually the process of learning, process of extending the knowledge that is already possessed.

In the "native" form of the knowledge in the human mind is interoperable, active and directly connected with the thinking process. It can be analyzed, modified, extended, adapted, connected with other knowledge – but all this process is limited to the mind of the particular human being. New contents of the knowledge (e.g. inventions) can be communicated to other people solely by encoding it again to the textual form (or speech). Only in this form it can be shared with others, e.g. discussed or published.

From the point of view of persistence and sharing with others, the "native" form of the knowledge is limited to the life of the person who possesses the knowledge (since it can exist only in his/her mind). Hence, so far the textual form is extremely important for humankind since it is the only way for the preservation of the knowledge for next generation and its extension (growing) by new findings, inventions etc.

We believe that the proposed model and environment will bring a new quality to knowledge management and use. It will mean a new kind of knowledge publishing, sharing, extending as well as preserving all knowledge for future.

Since the knowledge will be on Internet in an operable computational form, for many known applications the knowledge may be used directly and immediately, without the need to develop the particular software as we know it today. New kinds of applications will be invented which are impossible with today's approaches, technologies and with today's way of thinking about software, Internet and knowledge.

There are projects and approaches with a similar aim. Later in this paper we will mention some of them and compare their approach to that one proposed in this paper.

2.1. eThings - Representation of Concepts and Things

In general, knowledge is a model of a part of the real world (including abstract and virtual things, which never existed, however, which are already known). For this purpose knowledge management systems mostly use *concepts* (computing models, "classes", software) and *things* (concrete information and data, "instances" of the concepts, database). This dichotomy was adopted in most of the object-oriented programming languages, as well – classes vs. instances, albeit some OOPL use prototype-based model with direct empathy (inheritance) between objects.

We believe that the prototype-based approach better models the reality, i.e. there is no basic difference between concepts and things – both should be represented in the same way. The reason is that concepts evolved from things by extracting common features and they can be viewed as a kind of artificial and virtual things, which describe certain aspects of real things (take them "in front of the parenthesis"). Also, concepts act or are viewed as concrete things in certain situations (e. g. when being explored by scientists) and things can become concepts (becoming an example, prototype for subsequent similar things or even becoming a well-known notion - for instance Eiffel tower is still a thing, however, it has become prototype for many other towers as well as a symbol).

Hence, we propose *eThing* as a uniform representation for both concepts and things. eThing is an active object which lives and acts in the virtual world of WWM concurrently and jointly with all other eThings, just like real things (and concepts) in the real world. From the point of view of computer science we can say that eThings constitute a kind of a prototype-based object-oriented environment with active objects similar to actors or agents.

2.2. World-Wide Mind – A Cooperative Virtual World

Our aim is to describe the real world and knowledge about it by eThings. Hence, eThings will constitute a *virtual world* in which anything of the real world will have its place. For this purpose, eThings have to have features enabling to represent all aspects of real world things and knowledge – we will analyze and discuss these features later. If successful with this aim, the space of eThings will be a kind of virtual mind, similar to the mind of a man accommodating the knowledge in its "native" form. However, the difference to the human mind is that this virtual mind will be virtually unlimited in size, located on many computers on Internet, cooperative and shared by all users world-wide. From this reason let us call this space *World-Wide Mind* (WWM).

World-Wide Mind will be "cooperative" in two senses. Firstly, the conceptual contents of the model will be created and updated by a cooperative effort of many users worldwide. Internet and the technology of eThings will enable to share and reuse the results of the work of others. Conceptual eThings will constitute an ontology, which (from the taxonomical point of view) will be based on published and proven ontology systems. "Computational" part of the ontology will be defined in a way which will be close to the human thinking and natural language, quite neutral from the point of view of known "programming language". Thanks to these features, WWM will be *understandable* not only to programmers, but also (and first of all) to a much wider spectrum of Internet users – scientists, experts, students and users in various areas of interest, including social and economical sciences as well as diverse fields of technology. Understandability of the ontology and behavior description of WWM by a wide spectrum of Internet users is the basic assumption for their active cooperation on the development and use of the contents and hence it is the basic assumption of the acceptance and success of WWM as a whole. WWM will enable seamless cooperation of users world-wide who will be able to explore the published knowledge (use it for learning – "what are the things and how do they work"), use it for practical or professional purposes (use existing databases or build own ones, for public, closed groups or private, for fun or business etc.) or to develop and enhance the shared knowledge (researchers and professionals in particular domain).

Secondly, WWM will be "cooperative" for eThings between each other and between eThings and real world things (and people). eThings will live in the space of WWM and they will communicate and cooperate with other eThings as well as with things from the real world – they will be able to propagate information about events (in the real world or in WWM), request information or actions from other eThings or things and to react to events in an "intelligent" way. Under "intelligent" we mean that the behavior of eThings will be governed in a similar way as people understand that real world things behave: their action are based not only on their abilities ("methods") and explicit orders to perform them ("message send", "procedure calls"), but also by laws and rules, by spatial, topological and constructional dependencies on their neighbors and their parts, by unexpected events in their environment (independent actions of other things, accidents) and by time (unexpected events, passed chances etc.).

Actions of eThings may be performed also autonomously, initiated by themselves, caused e.g. by systematical evaluation of the state of their environment, anticipation of the future changes and finally a decision to act themselves, where the particular action will be selected from by some heuristic. Such behavior of eThings is a base for autonomic (self-regulating) behavior of WWM which might be one important aspect of WWM in the future (autonomic computing is a vision proposed by IBM for avoiding collapse of the exploding information technology in the future; it is inspired by the human body's self-regulating nervous system).

Such "autonomous" behavior should be considered much more as "pre-programmed" reaction of an eThing to the events in its environment than some kind of intelligence (thinking, intentionality). This "miracle" can happen solely because the way of the description of the behavior of eThings is similar to the way how people think of the behavior of the real world things.

Wim H.J. Feien [FEIE85] wrote in 1985: "Machines derive their usefulness from the fact that they do precisely what we instruct them to do. If we want to use them well, we have to learn how to conduct our reasoning power much more effectively, than traditional mathematics is able to supply." eThings with their features and functionality will be a fulfillment of this idea.

Hence, so far we cannot speak about any kind of "self-awareness" of eThings – they themselves have no mind or intelligence, their "intelligent" behavior is solely result of a good description of all eThings around them and their interoperability and cooperation. Using the six-level model of mental activities by Marvin Minski ([MINS06], p. 47), eThings will be able to perform instinctive reactions where the instinctive behavior is pre-programmed by constraints and rules. Later eThings will be probably to learn from the past and to perform learned reactions.

The cooperative effort of building and using WWM can be compared to Wikipedia – however, in contrary to Wikipedia, shared components of WWM will be a *live and working computing environment* (and knowledge management environment), not just (textual) descriptions. One can imagine it as an object-oriented programming environment (Smalltalk, Java) as well as all its applications including all "data", running on the Internet, however, with an object model suitable for modeling more aspects of the real world in an easier and better understandable way.

3. Separation of Concerns

The world exists and works in a certain way and people think of this world and its dynamics in a way, which reflects the world at a certain level of abstraction. The way how people think of the world is given not only by the being of the world, but also by the abilities of human mind to create and maintain mental images about the world (let's say: animals think in a different way and probably do not have these abilities).

In 1974, Edsger W. Dijkstra in his paper "On the role of scientific thought" [DIJK74] described his view of intelligent thinking. Here for the first time he mentioned the term "separation of concerns" when discussing the isolation of different aspects of a subject matter:

> "Let me try to explain to you, what to my taste is characteristic for all intelligent thinking. ... [snip]... It is what I sometimes have called "the separation of concerns", which, even if not perfectly possible, is yet the only available technique for effective ordering of one's thoughts, that I know of. This is what I mean by "focusing one's attention upon some aspect": it does not mean ignoring the other aspects, it is just doing justice to the fact that from this aspect's point of view, the other is irrelevant. It is being one- and multiple-track minded simultaneously."

Separation of concerns, i.e. breaking the matter of interest into distinct parts that overlap as little as possible is the basic principle not only of thinking, but of all programming languages and paradigms, as well. Particular concerns constitute atomic building blocks by which the whole is constructed. Different programming paradigms deploy different view when isolating concerns – for procedural and structured programming it is data structures, blocks and procedures, object-oriented programming deploys objects (classes) and methods as atomic behavioral constructs of objects. Encapsulation and information hiding are almost synonyms for separation of concerns. Separation of concerns is applied also in design patterns, frameworks, CSS (Cascading Style Sheets separating style from contents) and many other approaches or technologies in computer science.

The problem is that these approaches did not go far enough in the process of the identification and isolation of concerns. One reason is that most programming paradigms attempt to "stay pure" from the point of view of "their concern" and they do not allow to accommodate other aspects. Typical examples are functional and logical programming, but also structured programming, which isolates data from algorithms and procedures.

Another problem are cross-cutting concerns – functionality or non-functional requirements which cuts across methods or procedures (e.g. logging mechanism, persistence of objects, event driven functionality etc.). Answer to this problem is Aspect-Oriented Programming (AOP), which emerged out of object-oriented programming [KICZ97]. AOP defines *aspects* as stand-alone modules of functionality, which contain distinct concerns and which can join or fork at *join points*. The basic idea is good, however, deployment of AOP approach can lead to higher complexity of the whole program, unwanted functionality and unpredictable errors (e.g. renaming a function can lead to activation of aspects hidden somewhere deep in the program and to unwanted side effects; such errors are hard to find). Also, in our opinion the constructs defined by AOP are somehow "artificial" and they have little to do with the modeled reality (aspect, join point, pointcut, advice).

From the point of view of modeling the real world, object-oriented programming and associated approaches, methodologies and languages (e.g. UML) are most close to the way how people know and understand the world. Still, there are additional concerns which are not included in the object model of OOP – e.g. their spatial and topological features and others. Hence, let us discuss the "functionality of the world" in order to find relevant concerns that have to be included in eThing model as well as relations and dependencies between them.

To make it more simple, we will talk only about things and world – under "things" we will understand both things of the real world (with their features) and eThings of WWM, which should have similar features in order to model real world things.

4. Features of Things

4.1. Spacetime, Events and Identity

The world as we know it has 3 dimensions and time, its fourth dimension. Such a space is called *spacetime* or Minkowski space. This mathematical model of space and time as one space makes possible to describe spatial as well as dynamic aspects of things in both classical mechanics as well as in theory of relativity. For our purposes we consider Euclidean space and continuous time as forth dimension of spacetime.

Things are part of the world, in fact they constitute the world (otherwise it would be solely an empty space). There are only things of different kinds in the world and nothing else. Each thing has its place (location in the space) and may optionally accommodate some subspace (defined e.g. by its borders).

Things live in the world, the life of everything is a series of events, changes to the thing. An event is a point in spacetime specified by its time and place. Each event is unique – on one place at a certain time solely one event can happen, it is even not important, what had happened (what kind of change to which thing).

For each particular thing, the first event is its birth, which happened on some place at a certain time. Since each event is unique, two distinct things cannot be born at the same place and at the same time – even twins are born one after the other. Hence, the event of the birth best denotes the identity of the particular thing.

4.2. Action and Ability

Any change to a thing must be caused by some *action*, a change cannot happen from nothing, by itself. In each action, the thing performing the action is the main actor, it is "its action". To be able to perform any action, the thing has to have a certain *ability* (method). Each ability has a name which best describes the meaning of the particular action, change or event. This name is used for requests to the thing to perform the particular action (message send).

To perform an action takes some time – a non-zero time interval. Hence, there is not only one event associated with each action, but two – the start and the completion of the action. The action start denotes the event until which the thing was in a certain state and it was passive (not performing the action), between the start and the end of the action the thing is active, performing the action, which will probably cause some change to the thing (change its state) and the completion of the action is the event after which the thing is passive again and in the new state (including the change which had happened). These two events may be important for other things since they denote these changes to the things and hence the change to the whole system – other things may want to react to this change in a particular way (see later). Let us call these events *"before-action"* and *"after-action"*.

Changes in the world are unidirectional with time – they happen one after the other (or concurrently) and what had already happened, cannot be "undone". This is on of the fundamental principles of the world, time and life. Actually, the time is denoted by the changes – if there would be no changes, nothing could be observed in the system. From this reason people had invented clocks that generate "artificial" events e.g. every second, against which "normal" events can be compared and hence time can be measured.

4.3. State of a Thing – Attributes and Relations

The state of a thing is denoted by its *attributes* and by its *relations* to other things. Since everything in the world is a thing (and nothing else), also the value of an attribute must be some thing. However, attributes are somehow "more private" to the thing – in contrary, relations between things which are "more public" – known by both (or all) things participating in the relation.

Relations between things are more complex, they are not just links to other things. Things participate in relations with certain roles that may be named, the relation has its cardinality (1:1, 1:many, many:many etc.) and it may have other features (e.g. constraints). Simply, relations between things are also things that represent the relation between two or more things in its entirety.

Each attribute and relation of a thing has a name which best describes the meaning or role of the particular attribute or relation for the thing. This name is used in message sends when requesting the value of the particular attribute of relation.

4.4. Constraints and Rules

Any change to the system can be caused only by some action. But some changes are impossible because of higher laws, e.g. laws of the nature. Also under certain circumstances it is not possible to start the action – because of reasons outside of the thing (or the action), e.g. some conditions in the environment of the thing. This is a typical cross-cutting concern – the reasons are external to the action, but in most programming languages they are checked within the ability (method, procedure) and only if the particular logical conditions are satisfied, the action itself is performed. Often there are several independent conditions, which are "inherited" from different reasons – such conditions are even cross-cutting each other.

Therefore the execution of an action is determined by *constraints*. All relevant constraints must be fulfilled before the action is started, i.e. they are bound to the before-action event. There may be several constraints for the same action from different reasons.

Constraints are checked *before* the action is initiated. If any of the constraints is not fulfilled, the action cannot be started at all. In some software methodologies and systems, before-conditions and after-conditions are used. If the after-conditions are not fulfilled, the action (or transaction) is taken back (rolled back, undone, cancelled). In the real world such approach is impossible – if something had already happened, there is no way to take it back. This is the reason for all unwanted accidents – unfortunately, none of them can be "rolled back" in the real world where time (i.e. all events) flows only in one direction.

Similar to constraints from the point of view of tightness to the action but independence from the ability are *rules*. In the real world rules describe independent actions, which are initiated by a particular action. Such actions may be reactions to the particular action (i.e. post-actions), but also pre-actions (performed straight before the action is performed, e.g. in order to prepare the necessary conditions for smooth performance of the action) or when-actions (concurrent with the action). Hence, rules may be bound both to the before-action and after-action event.

Rules define independent actions – called also *side effect*, which are completely external to the particular action. They may, but need not to be initiated when the particular "before-action" and "after-action" event occurs and if some condition is fulfilled (ECA-rules – event-condition-action). Each of the rules is independent from the others, depending on external conditions another rules may apply (e.g. another environment because of the movement of the thing).

Constraints and rules are important in the real world (an in our model) because people think and act rather in laws and rules than in "pre-programmed procedures". The world is parallel, many events happen concurrently and depending on the conditions, different things may react in a their specific way. This may cause unpredictable results. It is no possible to describe such a complexity of actions, events and their dynamic dependencies by methods or procedures of standard programming languages.

4.5. Kind of Things and Inheritance

For many centuries people have tried to understand the world of the things around them, to understand themselves, to understand their understanding. They explored the nature as well as their mind and spirit, e. g. looking for similarities between different things. The results of their findings was knowledge concentrated in *concepts* – completely artificial abstract things, which existed solely in their minds and which described the common features of several things – the *kind* of the things. When doing the same proves with concepts (instead of "real" things), a hierarchy of concepts was created – a taxonomy. In the taxonomy concepts and things build a hierarchy of *inheritance* where more general concepts contain fewer features, which, however, are valid for a wider range of more concrete concepts and things.

Taxonomies describe a part of the world from one selected aspect – e. g. the classification of organisms or plants from the point of view of biology. Particular concepts, however, can be viewed and classified from several points of view, e. g. some plants from the point of view of agriculture or people from the economical (or financial business) aspect. Hence, a thing can belong to several concepts at the same time, to each of them from a different aspect. This means *multiple inheritance* between things. If we consider one *aspect* of a thing, i.e. one of its kinds based on one inheritance branch in multiple inheritance, the thing may have different features and behavior as the same thing from some other aspect.

Things evolve, they change their quality and kind during their life cycle. The most obvious is the evolution of a human being – after the birth he/she passes several phases of the childhood it becomes an adult. In each phase of his/her life, he/she had different features, abilities, different view of the world and different knowledge. While being a child he/she had to obey his/her parents who were responsible for him/her. Suddenly (when adult) he/she is responsible himself, he/she has to know the law and has to behave accordingly. Hence, the inheritance of things can change during their live, i.e. the *inheritance is dynamic*.

Different features are inherited in different way. Inheritance means sharing particular feature. Attributes and abilities are inherited in the first-found way as we know it from OOPL. However, constraints and rules are inherited in a different way – all of the inherited constraints apply to the particular action, not only the first one found along the inheritance link. Hence, constraints and rules are inherited in an additive way, all inherited features apply in the particular situation (i.e. all constraints must be met).

4.6. Space and Environment

The *world is spatial*, things are a part of the world, they have their place in the world. Most things accommodate some smaller or larger subspaces of the world. Subspaces of the things are defined by their borders which are 0 to 2-dimensional spheres in 3-dimenional space (points, lines and areas, where lines and areas may be closed, i. e. spherical) and one point belonging to the thing (e.g. the location of the thing).

Some things may overlap, they may "go thru each other" (e. g. radio transmitter coverage vs. administrative districts), they are independent from the point of view of their spatial properties as if they would live in different spaces. Some others cannot overlap – they are either joined (with a junction as border between them) or disjoined (not directly connected). Such things accommodate a common *environment*, in which they are exclusive with each other, i.e. the environment is a topological space for all of them.

Environment of other things is also a thing (since everything is a thing) – e. g. the town is environment for all its inhabitants, companies, their houses and headquarter, all necessary streets and roads etc. This thing may also have its environment, e. g. the environment for the town is the county or state.

A thing may be in several different environments at the same time and it can change the environment with time. Hence, the relation between the environments is also multiple dynamic inheritance. Due to this the thing can belong to several distinct topological spaces (as mentioned above).

Environment is for the thing more than just a space where it is located. Features of the environment may be inherited by the things located in this environment – this is first of all the case of constraints and rules, but in certain cases it may apply also to abilities and other features. Typical example are different laws valid in different states, e.g. the speed limit. When a thing (a car) crosses the border to another state, it has to change the speed to another speed limit. Example of inheritance of the ability can be the way of movement: an animal moves on land an in water in a different way which is given by the environment, in which it is just located.

Environment is also the space for acquaintance of things, communication between them and propagation of events. A thing cannot know everything from the whole world – it knows by name only things, which are in the same environments as it. Hence, the environment is the namespace for identification of things by name. Another things in another environments may have the same name – this is the normal ambiguity of the world. Also, a thing can react only to events that happened in its environment – it cannot react to something which happened on another side of the world because it cannot observe such event.

4.7. Whole and Parts

Some things are attached to other things, they are parts of another thing – the whole which is formed by all of them. E. g. machines consist from parts – parts are simple, but when connected to each other in a special way, a machine is formed which is a thing of another quality. The whole takes the space defined by the union of all its parts. When the whole moves or turns in the space, all parts "follow" it.

The whole is environment for its parts, but a tighter kind of environment: it not only provides space for the parts and propagates events (to let the parts to react independent), but it enforces some behavior to all of its parts. For example when the whole moves, i.e. changes its location, the same movement is enforced to all its parts. If a wheel in a machine turns, it is not only an event to which another wheels may react, but some of them must turn, as well. Functional dependencies of this kind have to be defined in each particular case.

5. Implementation and Applications

Implementation of this vision will be done in two steps. In the first phase, a desktop version of eThings and WWM will be implemented which can be used on a single computer. During this phase the proposed concepts as well as the first version of a working environment (albeit not networked) will be developed. This version will enable development of the contents – the eThing library – and export of the library or parts of it (environments) in a portable way (based on XML).

In the second phase the system will be published on Internet for public access and use. This phase will start the cooperative effort, which will lead to the development and use of the World-Wide Mind.

During the first phase also the scripting language will be developed. Code name for the language is *eTalk*. The basic principles for the syntax of eTalk are based on Smalltalk and SELF and it is the aim to design the language even closer to natural language in order to make it understandable to a large public. Beside this language, standard programming languages can be used for diverse applications and interconnections of WWM with existing systems.

As for the space for eThings we believe that geographic coordinate system with latitude and longitude and the elevation (so called 2,5 D) is more important for most of the applications for public use than real 3-dimensional space. For special purposes and alternate WWM with real 3-D spatial model may be implemented, or this can be an alternate (or additional) coordinate system within WWM.

WWM will enable development and immediate use of applications in a wide range of human activities. As most important applications we see all kinds of monitoring and hyperlinking eThings with real world objects, and applications with the need for modeling geographic and spatial interactions and dynamics. Typical examples are general and investment planning, EIA (Environmental Impact Assessment), traffic planning and control and similar.

6. Similar Projects and Approaches

There are several projects and approaches which have a similar aim. First of all it is all *knowledge management* projects including projects developing *ontology* or *controlled vocabulary*, e.g. OpenCyc/ResearchCyc, Dublin Core, GFO, WordNet and others. These projects are similar to our approach by the aim to develop the ontology of concepts, however, their aim is not to develop a working computational environment, which may be used for on-line use and applications. These projects will be the most important source for the development of WWM contents and eThings library.

Another similar project is *Wikipedia* the aim of which is to develop a *public encyclopedia* by a cooperative effort of users worldwide. Wikipedia contains textual descriptions of very many things and can be used as an important source of information for WWM. Similar to our approach is also the cooperative way of the content development of Wikipedia – in a similar way the development and use WWM will have to be organized.

Further, *semantic web* as an evolving extension of the World Wide Web is similar to WWM. The aim of semantic web is to include on WWW semantics which can be read and processed by machine (software agents) and hence easier found, selected, shared and integrated with other information. Enabling technologies for semantic web are Resource Description Framework (RDF), diverse XML-based formats and Web Ontology Language (OWL).

Ubiquitous computing (ubicomp) is an approach and aim where information processing should be integrated into real world objects and activities. We believe that eThings and WWM will facilitate ubiquitous computing by providing the enabling computational environment for object hyperlinking.

Autonomic computing is a vision proposed by IBM with the aim to develop self-managing computing systems. It is inspired by human body's self-regulating nervous system and the aim is to avoid collapse of Internet and computing systems because of the explosion of its complexity. WWM and eThings can provide one of the environments suitable to fulfill such goals.

7. Past and Future Work

Basic ideas presented in this paper are not novel. They had been developed, implemented and published by the author in late 80-ties as "p-object theory" and Spacetalk system (presented in live demonstration session on OOPSLA'89 in new Orleans). Since then, parts of the enabling technologies had been developed by the author within commercial and non-commercial projects (ArtBASE – object-oriented database and distributed Smalltalk, GIS library, namespaces for Smalltalk which had been adopted also by GNU Smalltalk and Squeak).

In late 90-ties several papers on this topic were published by the author ("thinking things") [MRAZ98]. The trial of the author to start the development of WWM these times failed because sufficient support for the project was not found.

Today we believe that several similar project converge to the original aims of the proposed approach and model. This makes the ideas of eThings and WWM and their usefulness for practical purposes and applications more understandable. This is why this project was restarted with the aim to be developed as a cooperative effort of Internet users worldwide.

References

[AGHA08] Gul Agha: Computing in pervasive cyberspace, Communications of the ACM, Volume 51, Number 1, January, 2008

[DIJK74] Edsger W.Dijkstra: On the role of scientific thought, paper Nr. 447 on http://www.cs.utexas.edu/users/EWD/index04xx.html

[FEIE85] Wim H.J. Feien: Tasting the flavour of programming, working paper WF066 on http://www.mathmeth.com/wf/wf0xx.shtml

[GUTI05] Gilberto Gutiérrez, Gonzalo Navarro, Andrea Rodríguez, Alejandro González, and José Orellana: A Spatio-Temporal Access Method based on Snapshots and Events, In: ACM-GIS'05, Proceedings of the 14th annual ACM international symposium on Advances in geographic information systems, Bremen, Germany, 2005

[KENN06] Mark Mc Kenney, Alejandro Pauly, Reasey Praing, Markus Schneider: Preserving Local Topological Relationships, In: ACM-GIS'06, Proceedings of the 14th annual ACM international symposium on Advances in geographic information systems, Arlington, Virginia, USA, 2006

[KICZ97] Gregor Kiczales et al.: Aspect–Oriented Programming, In: ECOOP'97 Proceedings

[KNOL06] Roman Knöll, Mira Mezini: Pegasus: first steps toward a naturalistic programming language, ACM OOPSLA'06 Proceedings, Portland, Oregon 2006

[MEYE00] Bertrand Meyer: Object-Oriented Software Construction, Prentice Hall PTR; 2nd edition, 2000

[MINS06] Marvin Minski: The Emotion Machine, Simon&Schuster, New York 2006

[MRAZ92] Jan Krc-Jediny, Augustin Mrazik: Object-Oriented Geometry and Graphics, OOPSLA'92 Tutorial Notes, Vancouver, Canada, 1992

[MRAZ93] Augustin Mrazik, Janka Kleinertova: Spacetalk: object-oriented environment and tools for building open object-based GIS environments, OOPS Messenger 1993

[MRAZ96] Augustin Mrazik, Marius Walliser, Rik Smoody: Independent Objects for Distributed Applications, Informatik 1/1996, SVI/FSI Zürich, 1996

[MRAZ98] Augustin Mrazik: Thinking Things - World-Wide Mind by Internet-based Objects, OOPSLA'98 Business Object Workshop

[PERR06] Matthew Perry, Farshad Hakimpour, Amit Sheth: Analyzing Theme, Space, and Time: An Ontology-based Approach, In: ACM-GIS'06, Proceedings of the 14th annual ACM international symposium on Advances in geographic information systems, Arlington, Virginia, USA, 2006

[RIEH01] Dirk Riehle, Steven Fraleigh, Dirk Bucka-Lassen, and Nosa Omorogbe. "The Architecture of a UML Virtual Machine." In Proceedings of the 2001 Conference on Object-Oriented Programming Systems, Languages, and Applications (OOPSLA '01). ACM Press, 2001. Page 327-341

[UNGA87] David Ungar, Randall B. Smith: SELF: The Power of Simplicity, OOPSLA '87 Conference Proceedings, pp. 227-241, Orlando, FL, October, 1987