Upgrades and new features in MDT PE 2.2 release

by Błażej Sytar, Sales & Support Leader, ASTEC

At the turn of March we have published the new version of Magik Development Tools Professional Edition. This version consists of a couple of interesting features which we would like to briefly present. Here they are:

• Displaying inheritance of classes in form of a graph – now you can display the inheritance of a class in a much more clear, graphical form. It lets you observe both subtypes and supertypes at once. You can also drag any class with a mouse to highlight its connections and place it differently for better transparency.
• Transmitting to any GIS Session – it allows you to transmit code chunks, methods, or even whole files to any predefined GIS Session. So now you can quickly transmit your code to sessions assigned to different projects without having to reassign them.
• Smart Caret Positioning – this feature moves the cursor in Magik Editor according to Magik code syntax rules. The mechanism recognizes identifiers, symbols etc. and places the cursor in the right place accordingly. It is enabled by default.
• Emacs-like Auto Indentation – we have improved this code formatting feature and allowed its customization, so your code will look the way you like it.
• Displaying logical structures in the Debugger – displays the logical contents of specific objects. It is very useful in case of collection objects, you can easily access their contents without having to look into their implementations.

And what about the future? We have plans for new features and upgrades for a couple of years ahead. Currently we are working on adding support for MUnits and enabling UML modeling.

Dear Readers,

Last year was full of challenges and successes and we hope that this one will be even better. We would like to welcome you this spring with the third issue of MDT Magazine. What is in it? First page presents the review of the newest version of MDT: MDT Professional Edition 2.2.

We recommend an article about Scrum project management with Rational Team Concert™ in MDT environment. You can learn how to create a project and how to manage it in accordance to Scrum methodology.

Next you can read more detailed description of MDT’s 2.2 new feature, which is displaying the inheritance of classes in a graph. Very interesting functionality that allows you to observe how a specific class is connected to other classes in both ways: up and down the hierarchy.

Later on you will find a short description of our new licensing model, floating. A list of advantages and technical details will surely convince you that this is the best licensing possible.

In the interview section Mr. Matts Isuls from National Land Survey of Finland shares with us his impression of MDT and explains why his company decided to equip developers with our tool. Last but not least we present an updated overview of MDT Trainings, arrangeable in ASTEC HQ as well as individually at your company’s site.

Best wishes,
MDT Team, ASTEC

inside...

Upgrades and new features in the MDT PE 2.2 release .................. 1
MDT and Rational Team Concert™ ........ 2
MDT 2.2 new feature: Type Hierarchy in form of a graph ......................... 4
Everyday work with MDT .................. 5
Closer look at Floating licensing ............ 5
MDT in National Land Survey of Finland - Interview with Mr. Matts Isuls ............................... 6
Choose the smart way to learn MDT ...... 7
**Introduction**

Everyday IT project management, regardless of the technique applied to implementation, requires a considerable amount of time. In order for project managers and developers to be able to perform duties in the most optimal way, the implementation of a specified methodology for project management is a good practice.

The following article describes the method of implementing a project in Smallworld™ GIS technology according to the rules of Scrum methodology while using MDT and Rational Team Concert (RTC) tools.

**About Scrum**

Scrum is one of iterative methodologies for project management, based on agile software development techniques. The characteristics of this methodology are the predefined roles:

- **Product Owner** – responsible for the product concepts and selection of appropriate items in next iterations.
- **Team** – mostly a group of 5-9 people. The Team is in charge of implementing product features. The Team is supposed to be self-organizing and self-led.
- **Scrum Master** – a person who ensures that the Scrum process is used as intended, is not the team leader but supports and supervises the team as well as checks the correctness of carrying out the process.

and artifacts:

- **Product Backlog** – contains product requirements, does not focus on low-level solutions but on the features the product is going to gain throughout the process of implementation.
- **Sprint Backlog** – contains the list of work the team must address during the next sprint.
- **Burndown Chart** – shows remaining work in the sprint. Updated every day, it gives a simple view of the sprint progress.
- **Impediments List** – is simply a set of tasks that the Scrum Master uses to track the impediments that need to be solved.

An additional concept related to the described methodology is ‘sprint’. It is a period of time – typically two to four weeks – during which the team creates a product that can be presented to the customer. Within one sprint, there are specific items, making up a whole of a given product version. An iteration, on the other hand, ends with the sprint review. A crucial feature of the methodology is that during the sprint planning meeting, the team members determine which backlog items they are able to complete in a given iteration, and record them in the sprint backlog. During a sprint, no one is allowed to change the sprint backlog, which means that the requirements are frozen for that sprint and the team is obliged to complete the items.

**About Rational Team Concert**

IBM® Rational® product family provides a full range of offerings to ensure software and systems delivery success through practical, reliable, and extensible components. Rational Team Concert is a tool based on the Jazz™ platform that is used for project management. Taking into consideration the architecture, the tool consists of two parts – Jazz Team Server and Rational Team Concert application. It can be installed as a separate product or as the extension of the existing Eclipse installation. The following crucial processes are being supported by RTC: process management, manage-
SCRUM

Creating Product Backlog

One of basic artifacts in Scrum is Product Backlog. In order to create it, choose New -> Plan from the File menu and set the following parameters - Project Area and Iteration. In the next step go to Advanced Options tab and choose Plan Type -> Product Backlog. One is able to determine other types of plans (eg. Sprint Backlog) in a similar way. The created Product Backlog, ready to be edited, should be displayed in the window. One of the available tabs is Planned Items. Here, with the use of the available options, one can add the to the aforementioned backlog. One can also browse the items, taking into consideration the chosen criteria. According to Scrum rules, the items available in Product Backlog are general items. Therefore, each one of them can be assigned to a specific iteration and user. After that, it is automatically deleted from the view. Using Plan option (see Figure 1.), one can present the items in a graphic way as well as manage them with the use of drag-and-drop method.

Reports in Scrum

Another option is the possibility of previewing the charts defined for this methodology. The number of available charts is considerable and each chart can be adjusted. Additionally, one can also create their own template. One of the basic charts is Burndown (Fig. 3) which shows the progress of the implementation (from Team Artifacts view, choose Reports -> Shared Reports -> Work Item -> Burndown). The chart's content is supposed to resemble the one form Figure 1.

If we are unsure what has been presented, when an unknown chart has been applied, the diagram enables the option: What does this report tell me? The description carries the information about what and in what way has been presented.

When working with the project, Work Items are basic functions. The very mechanism of creating is flexible about adjusting and enables changes of many factors. Work Items creating is intuitive, similar to other well-known systems of the kind. An important functionality is the items integration with the code. Rational Team Concert has its own system of versioning, but it is possible to use other solutions as well.

During item implementation, some fragments of the code are being changed frequently. Rational Team Concert makes it possible to connect fragments with the specific items. To do this, one should choose the specific item during change synchronization. In the first window one can type a regular comment. Next, the window allows to create a new item or choose the existing one from the list.

Figure 2. presents Work Item editor tab together with all changes in Change Explorer view that were assigned during code synchronization. While choosing a given element from a Change Set list, one can preview the changes.

Fig. 2 Work Item editor tab with task description and Change Explorer view with the list of changes in files

Fig. 3 Burndown Chart view generated by Rational Team Concert.

Rational Team Concert and MDT

Rational Team Concert tool has numerous functions that are useful when managing a project in any methodology. Apart from the above possibilities, building a version, synchronizing with many different software repositories, adjusting roles of users, and many more, are available. For MDT plug-in there is a possibility of such configuration of the aforementioned environment so that full capabilities of Magik project management are obtained. Thanks to that the implementation of, for example, Scrum, methodology is effortless, and, more importantly, the efficiency of team work is considerably improved.

Summary

Rational Team Concert, as a separate product, is an effective tool which significantly makes working on items easier. Nevertheless, in Smallworld™ GIS projects, where code development constitutes an integral part of items, it is a good idea to combine this tool with MDT. Thanks to applying the two combined products, it is possible to get a set in one ‘module’ which allows project development up to the present standards. As a result, both quick access to the code implementation with the use of MDT as well as an advanced interface for team work are reached.

www.mdt.net
MDT 2.2 new feature: Type Hierarchy in form of a graph

by Dariusz Michura, Project Manager, ASTEC

With the new MDT Professional Edition 2.2 the hierarchy view has been upgraded with new views and layouts. Before, it was only displaying class hierarchy in the form of a simple tree-like layout and you had to choose whether to observe the parents or the children (i.e. supertypes or subtypes). This form was incapable of displaying both of them at the same time because of multi-inheritance which is very common in Magik language.

Thanks to the new graphical layout not only the view is much more transparent and flexible but most importantly you can now observe the inheritance of subtypes and supertypes at the same time. The hierarchy view is now also accessible from almost every view and editor in MDT. Whenever you have a class or method name given you can quickly observe its hierarchy by selecting it and pressing F4 or choosing an appropriate action from its context menu.

The new graphical view is consistent with the UML diagram standards. It was based on an open source framework from Eclipse Community called Zest. It delivers three different types of layouts:

- **Tree layout** – default and most common view, compliant to UML Class Diagram,
- **Spring layout** – sets all connections to similar length, might be useful for classes with many elements,
- **Radial layout** – shows the main element in the center and its parents and children in concentric circles around it.

Graph is always displayed to best fit the window size, if you maximize the window the view will be refreshed and displayed accordingly. Each element can be dragged onto a different place with a mouse, you can also easily change the focus on to another class or open related source file by double-clicking on it. Selecting an element will highlight all connections to its direct parents and children so you can better identify the object itself as well as its placement in the class hierarchy.

Fig. 1 Standard layout (for subtypes)

Fig. 2 Standard layout (for supertypes)

Fig. 3 Spring layout

Fig. 4 Tree layout

Fig. 5 Radial layout
Everyday work with MDT

by Marek Wilniewiec
Software Developer, ASTEC

I am a young software developer with four years experience in Smallworld™. Before I started programming in Magik I was working with Java and C#. During my studies and further work I got used to programming in environments like: Netbeans, Eclipse and Visual Studio.

I have to admit that my first steps in Smallworld™ were very difficult. This was mainly caused by the fact that there was no integrated, user-friendly environment for Magik language. My colleagues and I were very happy when ASTEC – my employer, introduced to us a beta version of MDT. It felt very familiar from the start because it is based on Eclipse – the environment that we all know very well.

My work description generally covers bug fixing and porting. Big products, a lot of code to understand and a little refactoring are my primary tasks. I’ve been using MDT for over two years and my job has become much more enjoyable.

MDT has many useful features but the thing I like the most is using it in general. In my workspace there are only sources that I need and the way I left them the day before. I don’t have to search the whole hard drive each time I come to work. Everything is ready within two clicks: my session, all my sources and preferences. I don’t have to use external tools for searching or replacing. I really dislike the times when I need to do something in Smallworld™ and MDT is not installed on the computer. It is horrible when I want to find where a variable was initialized without Magik occurrences that highlight its usage. The same is when I jump over different methods, this is not so simple without outline and hyperlinks.

The things I mentioned are the things that we do in everyday work and MDT helped me to make it easier.

Closer look at Floating licensing

by Błażej Sytar
Sales & Support Leader, ASTEC

Why Floating?

Floating is one of the new licensing models in nowadays software market. It has a few substantial advantages that convinced us to include it in MDT licensing system. But what exactly is so special about floating?

We all know how it is with high quality business software and its licensing. Standard licenses, such as single user license assign the license to a specific user or a specific machine. It causes problems in case if you want to move the software user rights to someone else, if you have a habit of working on a couple of different machines or you just don’t want to purchase a license for every employee, because not all of them will use the software at the same time.

Floating handles these problems perfectly. Basically, the idea is to issue the license to any user that starts the application until there are available licenses left. If any active user closes the application his license becomes free for anyone else. And what if you have to occasionally leave your usual workplace and want to have your software with you? Floating also offers the ability to borrow the license from server, so you can work independent from server for a specific period of time (default 14 days, but it is of course customizable). When the borrow period ends the license becomes automatically unlocked on the server, but it can be always manually returned earlier.

How does it work?

MDT Floating License Server runs on LM-X software by X-Formation. It is additionally secured by a USB Dongle that confirms the validity of server software. Server controls the license traffic by issuing available licenses to every client that attempts to run MDT. If the number of user requests exceeds the number of available licenses, a license has to be freed in order to launch MDT on a different machine.

Setting up the server is very easy and takes approx. 5-10 min. Then you only have to enter given alias name, port and host address to each user’s MDT and you are ready to go.

Already using Floating:

- EPCOR Canada
- CIS CONSULT Germany
- EWZ Switzerland
- Ubisense Germany
- FirstEnergy USA
- NIS Switzerland
- NATIONAL LAND SURVEY OF FINLAND Finland
About one year ago, during GE Energy’s EMEAI conference in Nordwijk, Netherlands, we met Mr. Matts Isuls from the National Land Survey of Finland (NLSF). We spoke about the everyday challenges that he and his colleagues were going through while working with the Smallworld™ technology. He was very curious to see our new IDE for Magik, hoping it can ease his work. Not even a year has passed and the NLSF development team is already working with MDT. Now we asked him if he could spend some time with us again in order to answer some questions. He readily agreed and here are the fruits of this interview.

Mr. Isuls, you and your teammates are now working with MDT. How did it all start and what convinced you to the new environment?

NLSF started to renew the Cadastral System in 1994 with Smallworld™ GIS, it was version 2.2 and Emacs was the designated IDE for it. For most users the Emacs environment is hard to get into. Customizing Emacs is not the easiest task and then there are all those key-shortcuts. Generally software developers are familiar with Eclipse or similar IDEs like Visual Studio, this makes it much easier to get started with Smallworld™ development in MDT.

Besides MDT we still have several senior developers that use Emacs as their main development environment.

Could you share with our readers what has changed in your and your team’s everyday work?

The difference between MDT and Eclipse is minimal, it helps if you have to frequently switch between them. I believe it’s important when you can work in the same manner in both environments and don’t have to keep remembering how things work in a particular IDE.

Spelling errors occur less frequently with the help of Code Assist and errors in variable names are exposed easily while transmitting code. Selected word highlighting throughout the current source file is also a great feature compared to Emacs. Such things speed up development process when you don’t have to waste time on hunting down spelling errors.

Editing XML files and other tasks can easily be done with appropriate plug-ins.

Could you briefly explain what are the business benefits of this migration?

First of all, time saving during development. The Debugger is well made and has helped us in tricky situations. It’s much more enjoyable to work in MDT and it feels like you get more work done and have more control over the development.

In your opinion, what are the most important qualities of MDT?

Extendibility with different plug-ins.

Intuitive user interface and similarity to Eclipse.

Reliable and stable, has not crashed yet :-) 

Our goal is to provide fast and comprehensive support for the product. What is your experience in this matter?

We are very happy with the response time and result of the support. Problems we had were attended to and have been worked out fast.

In your opinion, what impact will MDT have on the Smallworld™ community?

A positive impact most likely. Modern development tools help the user to do a better job because you can concentrate on the task at hand while the tool takes care of careless writing errors. In addition modern tools like this one are more attractive to new and forthcoming developers.

How would you encourage other companies and teams to start investing their time in MDT methodology?

Development with MDT becomes easier and more pleasant thanks to code writing assistants and spelling checkers that leaves more room for productivity. Because of its flexibility and extendibility it can be used in different situations and even the picky users can easily modify it to their liking.

Thank you for the interview!

Mr. Isuls, you and your teammates are now working with MDT. How did it all start and what convinced you to the new environment?

NLSF started to renew the Cadastral System in 1994 with Smallworld™ GIS, it was version 2.2 and Emacs was the designated IDE for it. For most users the Emacs environment is hard to get into. Customizing Emacs is not the easiest task and then there are all those key-shortcuts. Generally software developers are familiar with Eclipse or similar IDEs like Visual Studio, this makes it much easier to get started with Smallworld™ development in MDT.

Besides MDT we still have several senior developers that use Emacs as their main development environment.

Could you share with our readers what has changed in your and your team’s everyday work?

The difference between MDT and Eclipse is minimal, it helps if you have to frequently switch between them. I believe it’s important when you can work in the same manner in both environments and don’t have to keep remembering how things work in a particular IDE.

Spelling errors occur less frequently with the help of Code Assist and errors in variable names are exposed easily while transmitting code. Selected word highlighting throughout the current source file is also a great feature compared to Emacs. Such things speed up development process when you don’t have to waste time on hunting down spelling errors.

Editing XML files and other tasks can easily be done with appropriate plug-ins.

Could you briefly explain what are the business benefits of this migration?

First of all, time saving during development. The Debugger is well made and has helped us in tricky situations. It’s much more enjoyable to work in MDT and it feels like you get more work done and have more control over the development.

In your opinion, what are the most important qualities of MDT?

Extendibility with different plug-ins.

Intuitive user interface and similarity to Eclipse.

Reliable and stable, has not crashed yet :-) 

Our goal is to provide fast and comprehensive support for the product. What is your experience in this matter?

We are very happy with the response time and result of the support. Problems we had were attended to and have been worked out fast.

In your opinion, what impact will MDT have on the Smallworld™ community?

A positive impact most likely. Modern development tools help the user to do a better job because you can concentrate on the task at hand while the tool takes care of careless writing errors. In addition modern tools like this one are more attractive to new and forthcoming developers.

How would you encourage other companies and teams to start investing their time in MDT methodology?

Development with MDT becomes easier and more pleasant thanks to code writing assistants and spelling checkers that leaves more room for productivity. Because of its flexibility and extendibility it can be used in different situations and even the picky users can easily modify it to their liking.

Thank you for the interview!
Choose the smart way to learn MDT

by Bartłomiej Łączkowski, Software Developer, ASTEC

In the modern competitive world, a growing number of employers require staff with high level skills. Users of programming tools gradually develop skills that enable them to perform their tasks more efficiently. Such process takes time, sometimes even years when there is no instruction. That is why teams often decide to make use of trainings to accelerate learning. Trainings entail more efficient programming, and less time spent reading manuals and making novice errors.

Thanks to the MDT trainings the participants will acquire knowledge how to easily switch from Emacs to MDT. The course shows how to properly configure and start the MDT environment. Trainees will also have an opportunity to familiarize with all of MDT functionalities and learn the best operation practices. During the training a wide variety of settings and wizards that accelerate developer’s work will be brought up. Advanced courses provide in-depth insight into MDT environment. They will show how to boost productivity using additional plug-ins and how to utilize MDT environment to its maximum potential.

The trainings are mainly intended for Magik developers. However, some elements can be very useful to project managers and software architects. The following trainings features instructor-led classroom teachings, labs and practical exercises.

Development techniques with the use of MDT

Development techniques with the use of MDT Development techniques with the use of MDT Development techniques with the use of MDT is a comprehensive 2-day training. It covers all aspects of working with MDT from simple basics to most advanced operating techniques. In the first part of the training, students will acquire knowledge about the most effective ways of creating and managing Smallworld™ sessions and projects. The participants will also learn how to migrate existing and complex configurations of the sessions from Emacs to MDT environment. Second part covers editing, browsing and working with the code as well as debugging it. The main goal is to learn how to use the crucial features of MDT in line with advanced development techniques to increase the efficiency of Magik code development. The contents of this training also include description of the best practices of using MDT as well as some tips & tricks which can boost the development process. Upon completion students will have a comprehensive knowledge about working with MDT and will be able to independently develop complex Smallworld™ applications from the beginning to the end.

Advanced aspects of MDT environment

Advanced aspects of MDT environment is an additional course considered as an extension to the 2-day comprehensive training. The main goal of this training is to present the possibilities of extending MDT development environment with the set of very useful plug-ins which can boost the productivity and speed up the development process. During this training the participants will learn how to create the multiplatform and integrated development environment based on Eclipse and MDT. Students will learn how to use the most interesting plug-ins which provide support for team work, task-oriented development, issue tracking systems, other programming languages like Java and some other useful tools.

MDT Magazine

Publisher ASTEC Sp. z o.o. © 2011 ASTEC Sp. z o.o. All rights reserved.

MDT is a product of

ul. Piaskowa 14
Pl. 65-209 Zielona Góra
Phone: +48 68 32 980 00
Fax: +48 68 32 980 03
info@astec.net
www.astec.net

Visit our website: www.mdt.net

Trademarks and registered trademarks. MDT Magik Development Tools is a registered trademark of ASTEC Sp. z o.o. Smallworld is a registered trademark of General Electric Company. Eclipse is a registered trademark of Eclipse Foundation, Inc. Other company or product names mentioned in this publication may be trademarks or registered trademarks of their respective companies.
Powerful solutions through advanced technologies.

As a strong GE Energy authorized Solutions Provider we offer software development services for Smallworld™ users from utilities and telecommunication industry sectors.