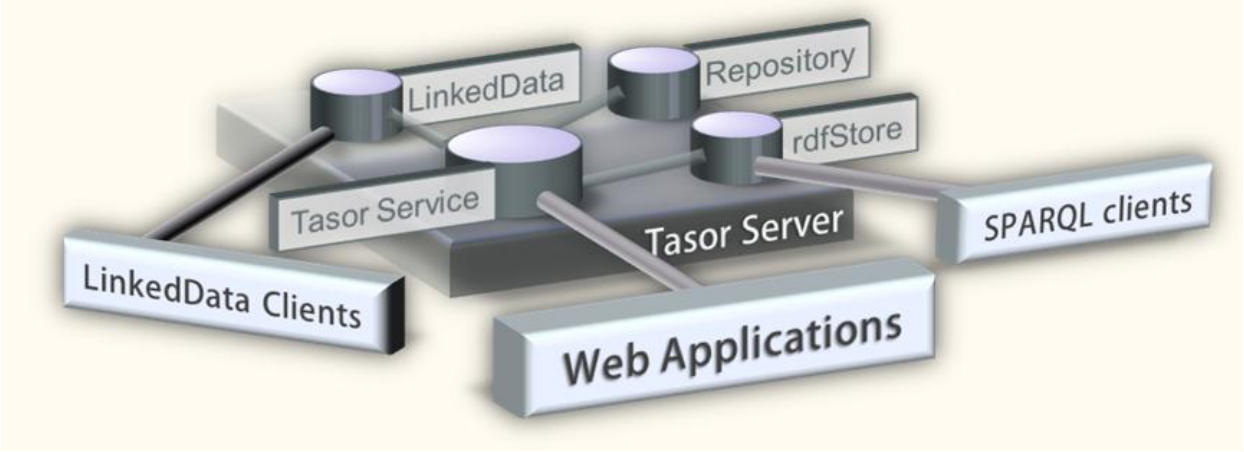
TASOR

SMART CONTENT APPLICATION SERVER



VIRTUONA

TASOR SMART CONTENT APPLICATION SERVER

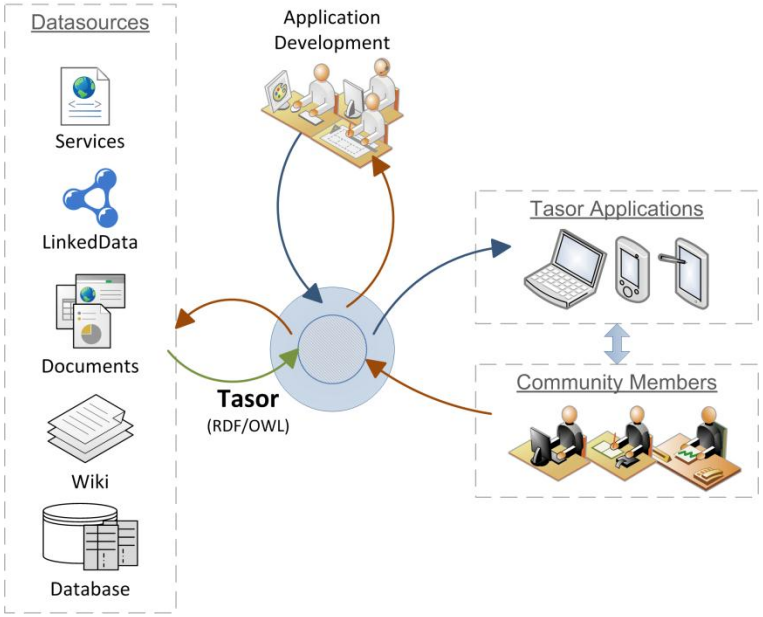


Figure 1 - Tasor Platform

Tasor server can take data from any source. You can create new data from scratch, or based on existing data. Newly created data can be stored locally or be published on the web to be used be yours or any other application. Data flow is represented with green and brown arrows on the picture.

Your development team can create application based on this data. This means that no other data manipulation is needed before data are showed to user (every data manipulation is done automatically, based on the rules that you define inside the data).

Applications that are created based on Tasor architecture are simple HTML applications that can be accessed from any device. Of course HTML applications are not the only one that can work with Tasor server, you can develop other server application that will communicate with Tasor, or some desktop application, plugins for other applications etc.

ARCHITECTURE

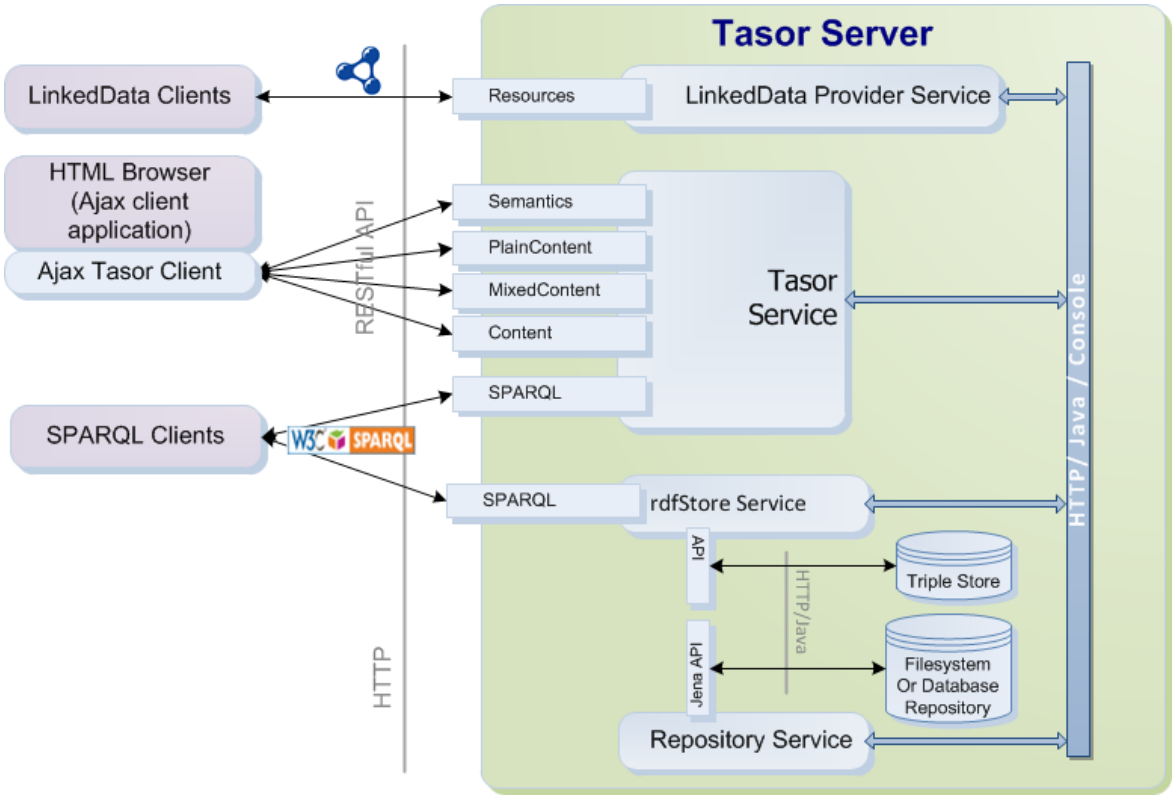


Figure 2 - Tasor Server architecture

Tasor server consists of number of services. This mean that its scalable and distributed architecture will handle any number of users and data. Communication to those services is done in RESTful way, over HTTP, with security protocols. This enables you to communicate with services from any other application.

Main service in Tasor server is Tasor service. Its job is to represent semantic resources in a way that is easy to work with. On the backend it uses large distributed semantic triple store repository. To the client applications it provides easy API to work with this repository. If you prefer you can access semantic store directly via SPARQL endpoint, or you can use public Linked Data API for accessing and publishing semantic resources.

On the client side we provide JavaScript libraries for building rich internet applications by communicating with Tasor server via AJAX technology. This application can take full benefit

of semantic web technologies, and provide you an easy way to deliver rich and intelligent client application in any Web browser.

SERVICES

On the server everything is service. You can use our services, create your own, or connect to external ones. We provide basic libraries for fast development of secure Java services. Every service can be run on any machine, and any number of times. Any service that is available in the system can be monitored, accessed, stop in an easy way.

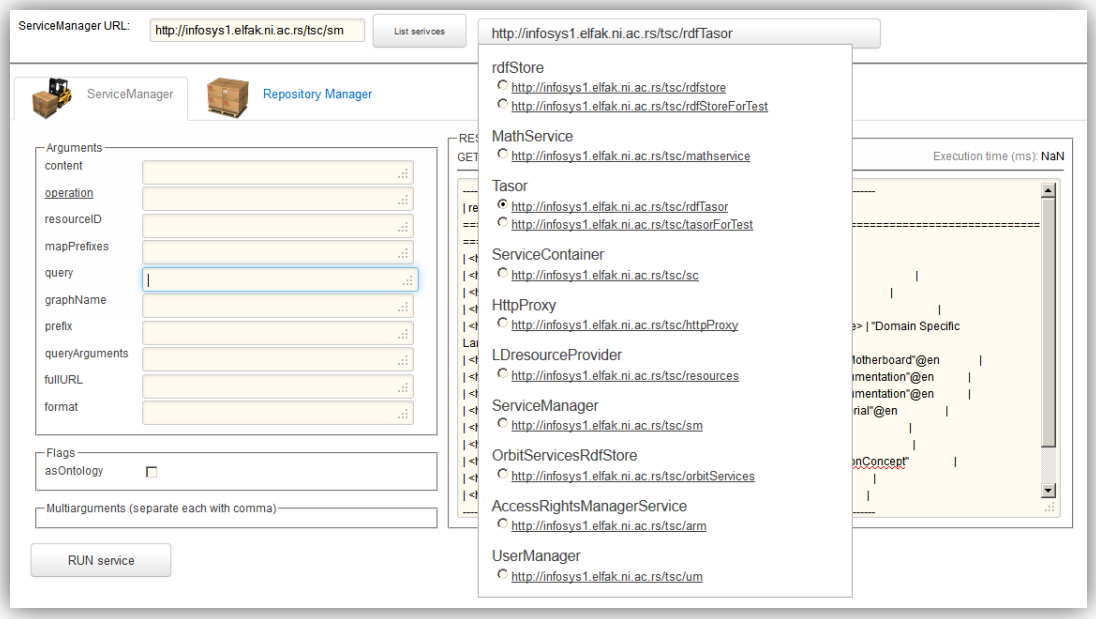


Figure 3 - Tasor Service Manager

Services that are currently developed, and can be used by any client application, are:

- rdfStore service – repository for semantic data. Provides access for executing SPARQL queries for creating and retrieving data.
- Repository service – can be used as content repository for any type of resource. Has ability to group resources in collection and can have multiple backend implementation for storing binary content of resources.
- Tasor service – this is main service that is used to interpret client request in appropriate way and respond to them by calling other services. It has well defined API that hides all complex things that are ongoing under it.

- Http Proxy service – utility service that can be used by HTTP clients to overcome cross domain problem.
- LDresource provider service – simple service that can be used as external access point for other LinkedData systems, so the data from Tasor server can be access in standard way.
- Service Manager service – every service in the system is registered to this service. It contains information about whole Tasor server and its configuration. By using this service you can see what is going on in the system.
- XML2RDF service – this is tool for converting XML descriptions of various external services to RDF format, so they can be imported in Tasor server.
- User Manager Service – this service is responsible for registering, log in and out of users. It also contains information for those users. Beside this, this service is capable of managing groups for users.
- Access Rights Manager service – does management of access rights for any service that has them. Service describes their actions and default access rights to this service. For any user or group of users in the system you can set access rights via this service. When user tries to execute some action in the system then this action is first checked with this service, if user has right to execute requested action, then action is performed, else exception can be thrown.

Services are implemented in Java. Every service can be called as instance from within Java code, or it can be called via HTTP protocol. Services that can be accessed with HTTP are wrapped in Java Servlet technology. This way we have various methods for instantiating services and executing their methods. From Java code there is no difference which way service is called.

Whole architecture is developed from scratch and can be used for developing any other services. By using it you gain ability to interconnect various services, have access right and security mechanisms, work with users, deploy services on any machine, manage services in your system etc.

DATA

Resource is the basic and atomic unit of data in Tasor. Resources are abstract concepts and can represent anything that exists in physical or mental world. Resources are connected among them by statements. For any concept that exists in knowledge that you are describing you should create resource.

Statement represent triplet of resources in which the first one is called Subject, second one is called Predicate, and the third one is called Object. By creating statements you are describing resource and adding their existential restrictions.

Dataset is a special kind of a resource, it is a collection of other resources and their statements. By grouping resources in one dataset you are able to separate your knowledge from other ones. You can export/import yours datasets, you can set access rights on them, and you can invite others to join you in describing you knowledge.

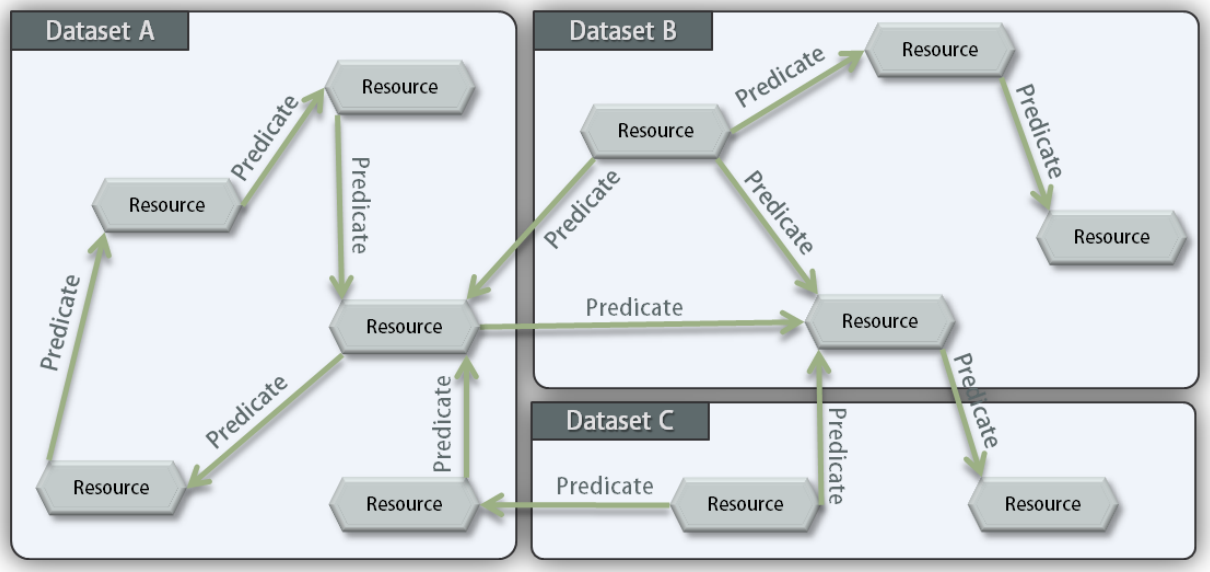


Figure 4 - Organization of data inside Tasor

There are special types of resources, the ones that have predefined behavior in Tasor applications. Those types are defined by the Semantic Web specifications, such as RDF, RDF/S and OWL. Inside Tasor you can retrieve information about resources in multiple ways. You can retrieve statements about resources, description of resources (in various data formats), or you can retrieve binary representation of one resource.

Querying the data is also available. There are two types of queries that can be performed on data. You can execute a number of predefined queries that represent most common requests. Or you can write your own queries to retrieve any type of information. Queries are written in SPARQL language and their results can be in various formats.

Changing data in Tasor server is also done via queries. There are a few things that you should have in mind when executing such queries. You can only change data in dataspace on which you have access rights to perform such action. A valid SPARQL update query must be

written for your request, or you can parameterize one of the existing queries. Changes on resources are visible immediately after executing a query.

By creating data in Tasor you can chose to make them public to community. This way others can view and use your data. This is done by managing access right on your datasets. Public datasets can be included in other datasets, access by the whole web (users and applications). You can also choose which users will be able to change your data.

Technical Info

For storing data about resources Tasor communicate with his rdfStore service (which has TDB semantic store at his backend). This service can be distributed on any number of machines so the amount of data that can be stored in it can be large. It also provides a valid SPARQL endpoint for any other client to access data in standard Semantic Web way. It can be also used.

LinkedData provider service is the one responsible for making data publicly available, so it can be reached from other Linked and Open data repositories.

For storing binary data of resource Tasor uses repository service. Same as other services and this one can be distributed over multiple machines to provide storage for vast amount of data.

APPLICATIONS

Tasor applications are Web applications that can be run on almost any browser (from any device). They communicate with Tasor server over HTTP, in asynchronous way. Application retrieves data from Tasor server, but application itself can resides on any machine.

Application is build using simple HTML, JavaScript and CSS technology. JavaScript is responsible for communication with Tasor Server. By calling Tasor API client applications can retrieve data from server. This data can be presented to the user or it can be used by application itself. Which data is presented to the user is defined by application. Users and applications can chose to filter data by datasets. Each application can be connected to one Tasor server instance at the time, but different Tasor server instances can be used as data sources by same application.

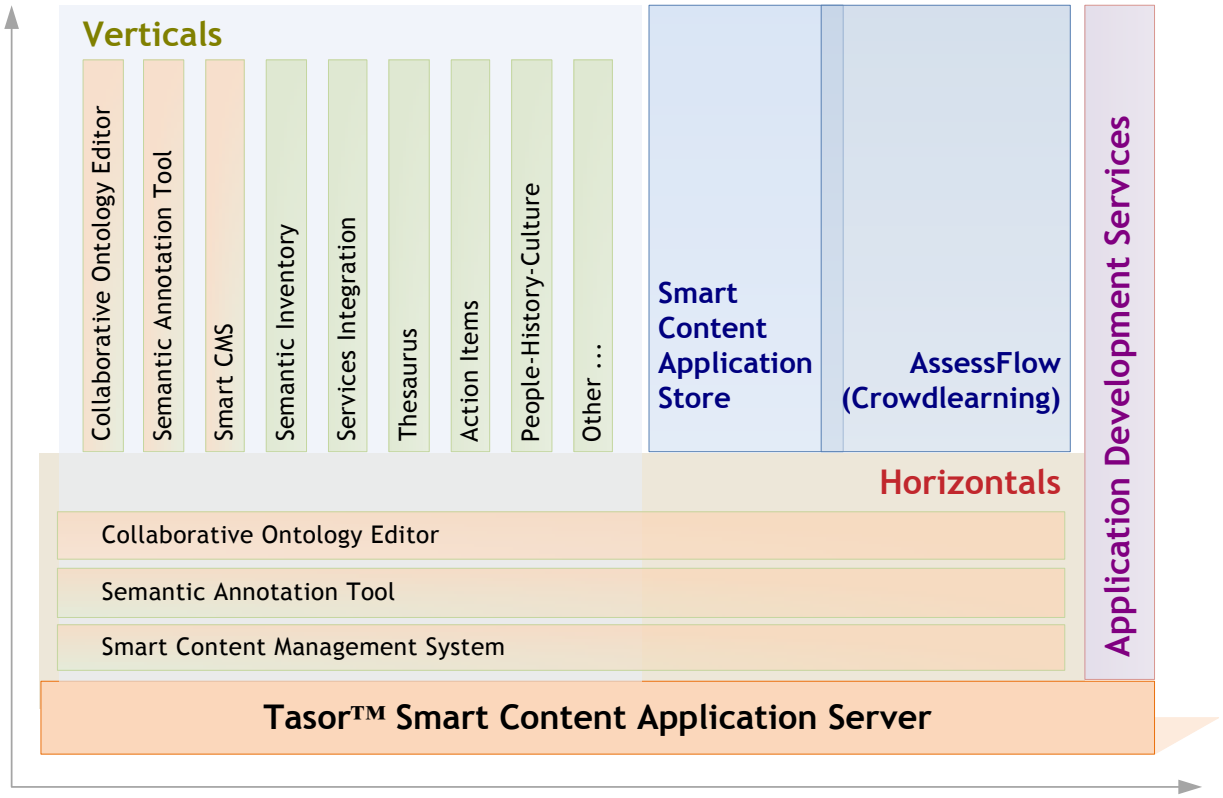


Figure 5 - Tasor Tools and applications

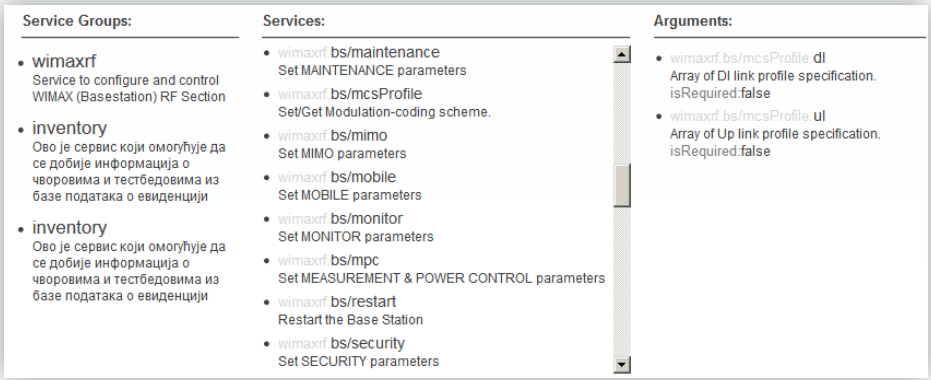
One of the strengths of Tasor server lies in fast application development. Rich Internet Applications (RIA) can be developed to provide users with tools to manage their data. Some of the applications to achieve this goal are standard and are already available in Tasor server.

In the foundation of Tasor server are following applications:

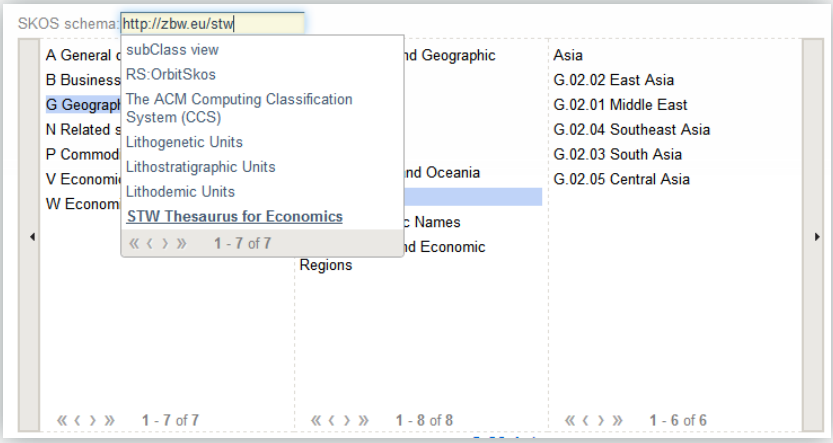
- Smart Content Management System – tool for collecting, defining, and accessing all of yours data sources.
- Semantic Annotation Tool – tool for describing data that you will use. How they are interconnected, what they have in common and how will they be represented in you applications
- Collaborative Ontology Editor – tool for working with your metadata. Domain experts can define data structures and reasoning rules on this level, in collaborative, secure, correct and easy way.

To help you in development of smart content applications we have provide several tools that you can use:

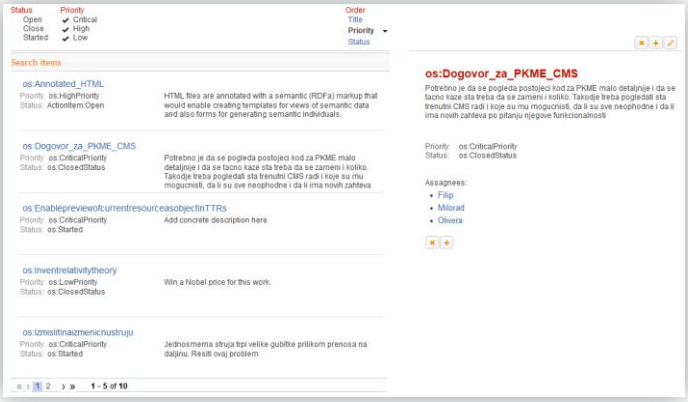
- Semantic Inventory - ?
- Service Integration – for monitoring, integrating, describing and importing external services in your application



- Thesaurus – will help you to organize standard language that is used in your domain



- Action Items – you can use it to track all things that need to be done in your application and to inform users about ones that have been done



- People-History-Culture -? (you can use this application to getter information about your users, their affiliations, what they have been doing in your application etc)
- Other… - more application are to be developed by our team, and they will be deployed to your installation.

Technical Info

We have built JavaScript library that is used in our application. It is built using jQuery, and everything is based on its widget factory (to insure reusability). Communication to server is implemented using Ajax technology. Beside call to Tasor server this library has various utilities for helping in application development. HTML code is done in version 4 without using new features in HTML 5. CSS version that is used is 3. Templating mechanism is done by retrieving binary content of resources, the one that contains HTML markup for applications.

SECURITY

Communication with Tasor server has several levels of security. On the communication level we use HTTPS protocol which protects transfer of data between services and clients. Once the user is successfully registered with Tasor server all services and applications in the system are aware of his existence. When user submits his login information to Tasor server he can work with those credentials in any application.

Every action that user performs on Tasor applications and services goes under authorization checking. Tasor has very simple, yet powerful, authorization mechanism that can handle users, user groups, resources, resources types, actions and action groups in any combination. This enables you to create very complex authorization configuration in an easy way.

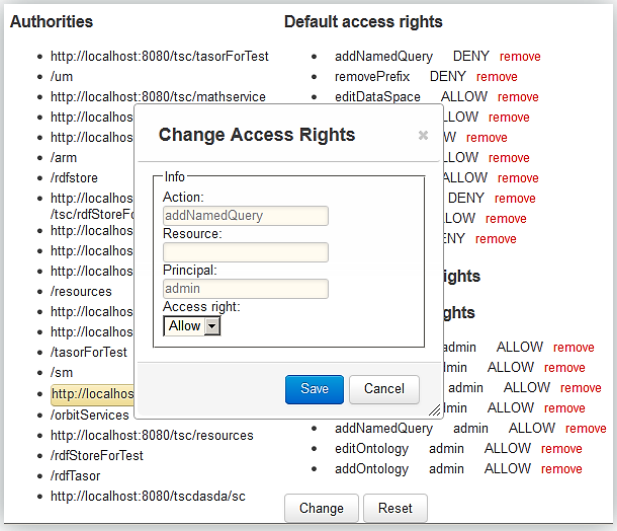


Figure 6 - Access Right Manager

Access Right Manager – ARM

ARM is our web application that can be used as a single point for setting up access rights for yours users. You simply select what type of action you want to permit, or deny, to some users, or user groups, and the whole system will began to obey those rules momentarily. Beside users and actions you can even set access right on the single resource, so you can have control on every aspect of your system. Access right configuration that you create can be saved for backup and restore later when it's needed.

USERS

Tasor enables your users to be registered and logged in from any application in the system. Registration process includes email confirmation, registration by invitation and secure store for user information. Users can belong to users groups, for easy user access rights management. All changes that you perform on users and their groups will be propagated to every service in the system, and changes will be applied in real time.

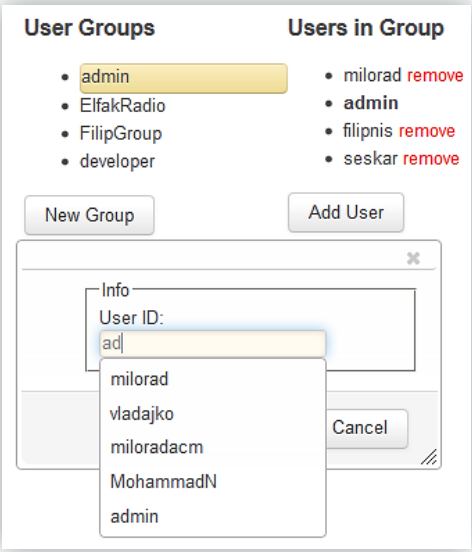


Figure 7 - User Manager

User Manager – UM

UM is web application that can be found in every Tasor server. It provides you with basic operations for user management. You can create groups, add/remove users from groups, set group admin, view access rights for users/groups and send notifications to users/groups. Who can make this changes is also define in ARM application.

DEVELOPMENT

For developing client applications we provide a set of components and libraries that are created to communicate with Tasor server. Currently we have developed a few mechanisms for Web Browser based application, but any client technology can be used the same way.

You can chose from many our components or develop your own. We provide components for searching, connecting, sorting, ordering, navigating, viewing, editing, creating, and deleting resources.

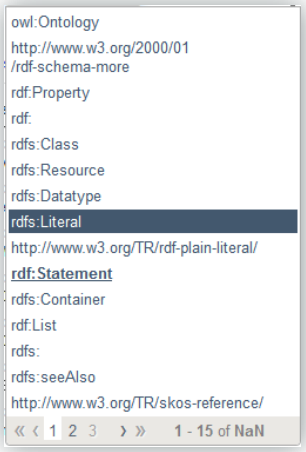


Figure 8 - List of resources

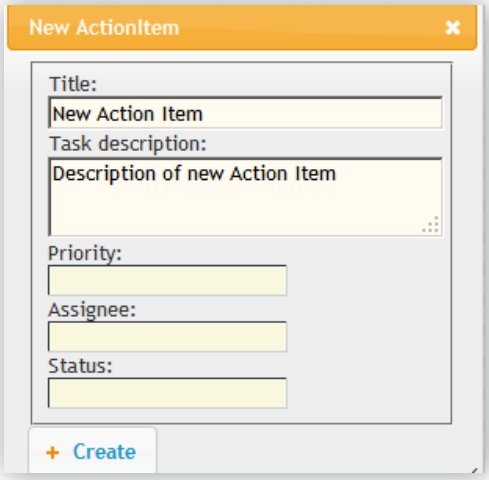


Figure 9 - New resource creation

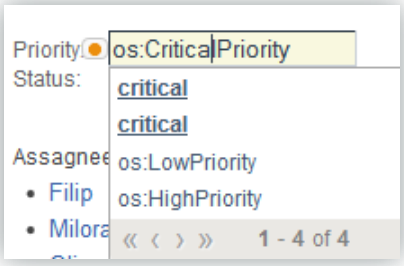


Figure 10 - Inline selection

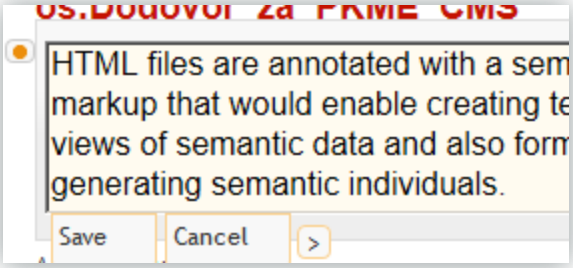


Figure 11 - Inline editing