## Runtime evaluation

### Metrics

Device side Hyperty Runtime (browser and standalone runtime) as well as Server side Runtime (NodeJS runtime) were.

#### Functional Metrics

The following functional metrics are used to evaluate the Hyperty Runtime in the Browser including:

Hyperty is correctly deployed in the Runtime including:

* Address Allocation
* Registration in the Domain Registry
* Associated with user Identity

Protostub is correctly deployed in the Runtime including:

* Address Allocation
* Connected to the Message Node

Idp Proxy is correctly deployed in the Runtime including:

* Address Allocation
* Connected to the IDP

Messages are correctly delivered by the Runtime:

* Among Hyperties in the same runtime (intra-runtime)
* Among Hyperties in different runtimes, from the same domain (intra-domain)
* Among Hyperties in different runtimes, from different domains (inter-domain)

Identity Management

* User is able to select IDP to be used independently of the Hyperty
* User is able to login with selected IDP
* Identity Assertions are correctly generated and added to the body of outgoing Messages
* Identity Assertions contained in the body of incoming messages are correctly validated by the IDP
* Communication between two hyperties is encrypted with tokens generated from the IdP Proxies.

Data Synchronisation

* Reporter creates a new object and invites one or more Observers
* Observer subscribes an existing object
* The object handled by the Observer is synchronised with the changes performed by the Reporter

Address reusage and data objects resume \* Address allocated to hyperty is reused between sessions \* Address allocated to data objects are reused between sessions \* Data Objects created by Reporters are resumed between sessions \* Data Objects subscribed by Observers are resumed between sessions

#### Non-Functional Metrics

For the Browser runtime, the execution time is the only non-functional metric used, for all functionalities described above.

For the NodeJS runtime the following non-functional metrics are used:

**Deployment Performance**

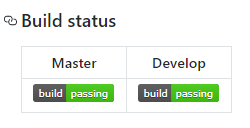
1. Number of Hyperties deployed / sec
2. Number of Protostubs deployed / sec

**Reporte-Observer data synchronisation performance**

1. Number of Data Objects created / sec
2. Number of Data Objects created / sec with data object children
3. Number of Data Objects subscribed / sec
4. Number of Data Objects subscribed / sec with data object children
5. Number of Subscribers per Data Object
6. Number of Subscribers per Data Object with data object children
7. Number of Data Objects changed / sec with different number of subscribers and size of the change
8. Number of Data Objects Childs created / sec with different number of subscribers and size of the data
9. Number of Reporter data objects resumed / sec
10. Number of Reporter parent data objects resumed / sec with different numbers of data object childs
11. Number of Observer data objects resumed / sec
12. Number of Observer parent data objects resumed / sec with different numbers of data object childs

### Browser runtime Tests

The evaluation of the Hyperty Runtime is performed with Karma Unit tests implemented [here](https://github.com/reTHINK-project/dev-runtime-core/tree/master/test), and for each one the execution time was measured as defined above. For quality control purposes, these tests are automatically executed every time a git push or git pull request is performed to develop and master branch and its status is depicted following best Continuous Delivery practices. The quality status is depicted in the Github repository *readme* page:



Runtime Quality Status

The full detailed report of the tests execution are provided below where we can see that:

* Hyperties are correctly deployed in the Runtime including the allocation of addresses, the registration in the Domain Registry and the association of User Identities;
* Protostubs are correctly deployed in the Runtime including Protocol Stub is successfully retrieved from the Catalogue server, an Hyperty Runtime URL is successfully allocated to the protocol stub, and the protocol stub is instantiated and connected to the Message Node.
* Idp Proxy is correctly deployed in the Runtime that is used to authenticate user with selected IdP e.g. Google IdP. The IDP Proxy is successfully retrieved from the Catalogue server, an Hyperty Runtime URL is successfully allocated to the IDP Proxy, and the IDP Proxy is instantiated and successfully connected to Google IDP.
* Messages are correctly delivered by the Runtime including:
* messages were successfully exchanged among Hyperties running in different runtimes, from the same domain (intra-domain).
* messages were successfully exchanged among two Hyperties running in different runtimes, from different domains (inter-domain).
* User is able to select IDP to be used independently of the Hyperty
* The user successfully authenticates with selected IdP.
* Identity Assertions are correctly generated and added to the body of outgoing Messages
* Identity Assertions contained in the body of incoming messages are correctly validated by the IDP

*TODO: add short performance analysis*

*TODO: insert karma report here*