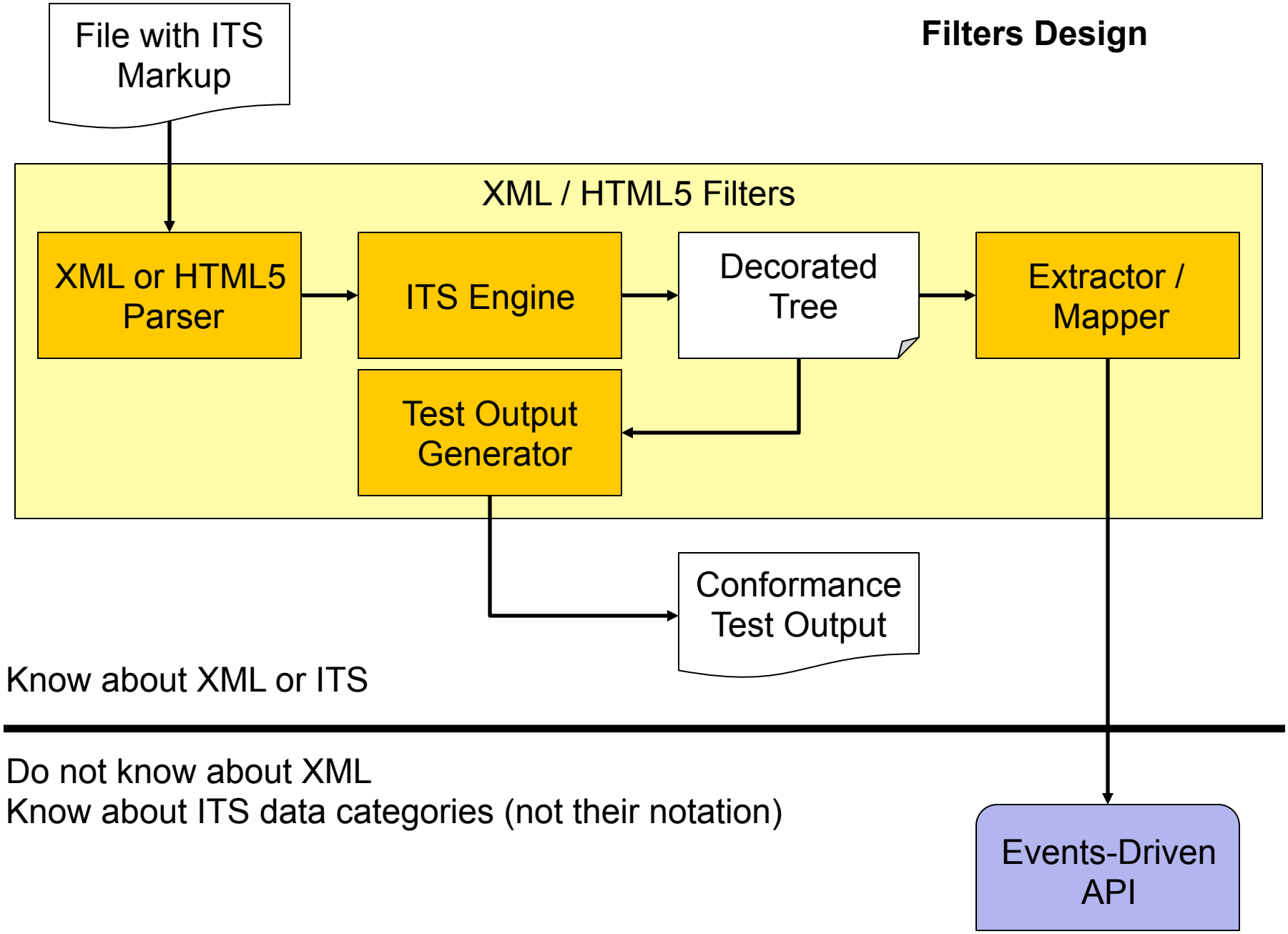
Some Use Cases with the Current Okapi Framework Implementation of ITS 2.0

Prague – September 2012

Use Cases

- Simple Machine Translation
  (using Rainbow)
- Translation Package Creation
  (using Rainbow)
- Moses Translation (M4Loc, sort of…)
  (using Tikal, a command-line tool)
- Quality Check
  (using CheckMate)

Filters Design



Simple Machine Translation

Description

- XML and HTML5 documents are translated using a machine translation system, such as Microsoft Translator.
- The documents are extracted based on their ITS properties and the extracted content is send to the translation server. The translated content is then merged back into its original XML or HTML5 format.

Data Categories

- Translate
- Locale Filter
- Element Within Text
- Preserve Space
- (Domain)

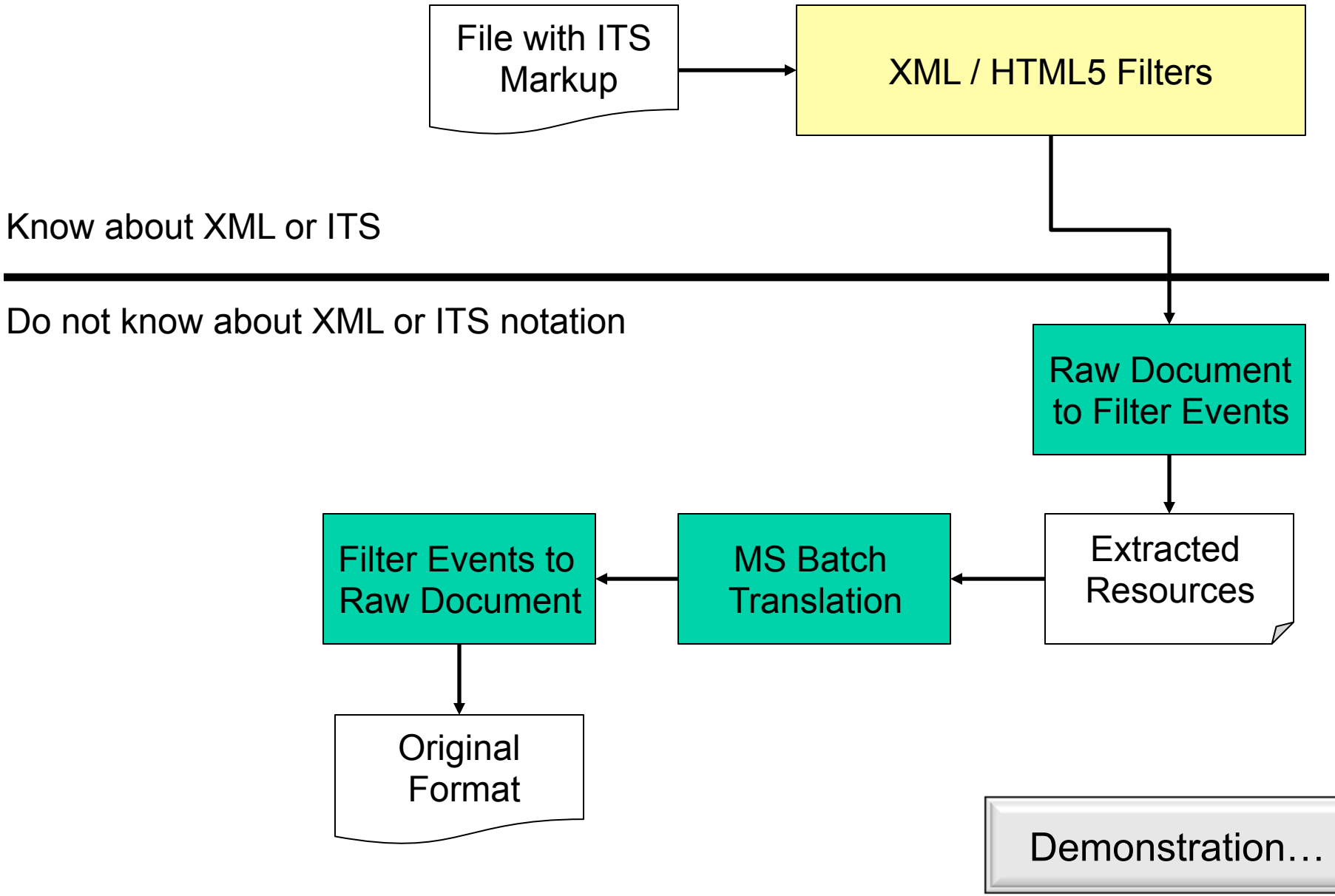
Benefits

- The ITS markup provides the key information that drives the extraction in both XML and HTML5.
- Information such as preserving white space can also be passed on to the extracted content and insure a better output.

Simple Machine Translation

- **Translate** - The non-translatable content is protected.
- **Locale Filter** - Only the parts in the scope of the locale filter are extracted, the others are treated as 'do not translate' content.
- **Element Within Text** - The information is used to decide what elements are extracted as in-line codes and sub-flows.
- **Preserve Space** - The information is passed on to the extracted text unit.
- **(Domain)** - The domain values are placed into a property that can be used to select an MT engine.

Simple Machine Translation



Translation Package Creation

Description

- XML and HTML5 documents are extracted into a translation package based on XLIFF.
- The documents are extracted based on their ITS properties. The extracted content goes through various preparation steps and save into an XLIFF package. The ITS metadata passed on and carried by the extracted content are used by some steps.

Data Categories

- Translate
- Locale Filter
- Element Within Text
- Preserve Space
- Id Value
- Domain
- Storage Size
- External Resource
- Terminology
- Localization Note
- Allowed Characters

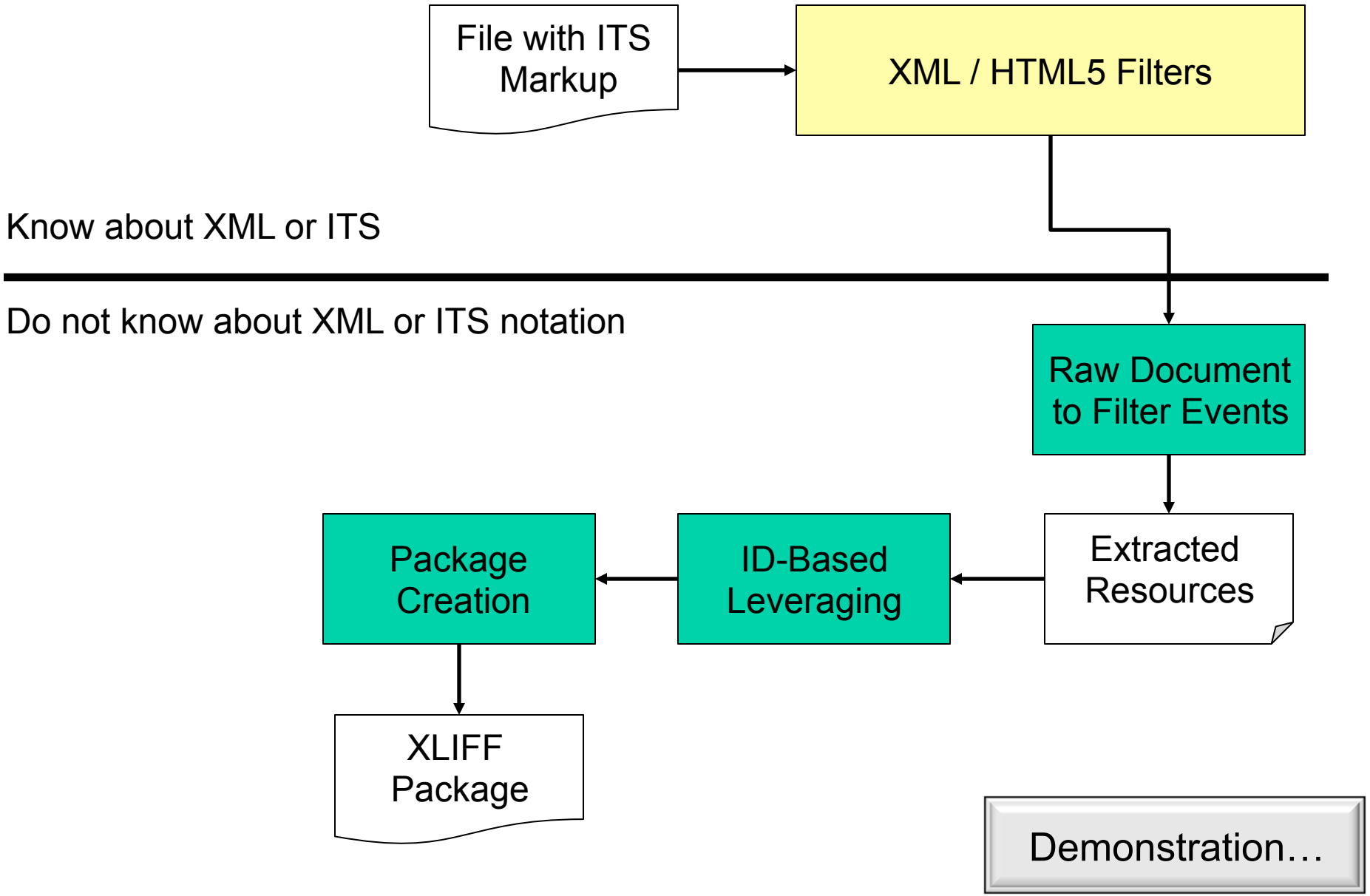
Benefits

- The ITS markup provide the key information that drives he extraction in both XML and HTML5.
- The documents to localize can be compared against older version of the same documents using ID to retrieve match the entries, and existing translations can be retrieved automatically.
- Information such as the domain of the content, external references, localization notes are available in the XLIFF document so any tool can make use of them to provide various translation assistance.
- Terms in the source content are identified and can be matched against a terminology database.
- Constraints about storage size and allowed characters can be verified directly by the translators as they work.

Translation Package Creation

- **Translate** - The non-translatable content is protected.
- **Locale Filter** - Only the parts in the scope of the locale filter are extracted, the others are treated as 'do not translate' content.
- **Element Within Text** - The information is used to decide what elements are extracted as in-line codes and sub-flows.
- **Preserve Space** - The information is mapped to xml:space.
- **Id Value** – The value is mapped to the name of the extracted text unit.
- **Domain** – The values are placed into an <okp:itsDomains> element.
- **Storage Size** – The size is placed in maxbytes, and the native ITS markup is used for the other properties.
- **External Resource** - The URI is placed in a okp:itsExternalResource attribute.
- **Terminology** - The terminology information is placed into a specialized XLIFF note element.
- **Localization Note** - The text is placed into an XLIFF note.
- **Allowed Characters** - The pattern is placed in its:allowedCharacters.

Translation Package Creation



Moses Translation (M4Loc)

Description

- XMl and HTML5 documents are translated using Moses through the M4Loc scripts.
  Note: In this demo we use sed instead of M4Loc scripts
- The documents are extracted based on their ITS properties by Tikal and converted into an intermediate format. The temporary files are run through the translation process. Tikal is then used again to create a translated version of the XML and HTML5 documents based on the original source documents and the translated intermediate files.

Data Categories

- Translate
- Locale Filter
- Element Within Text
- Preserve Space
- Domain

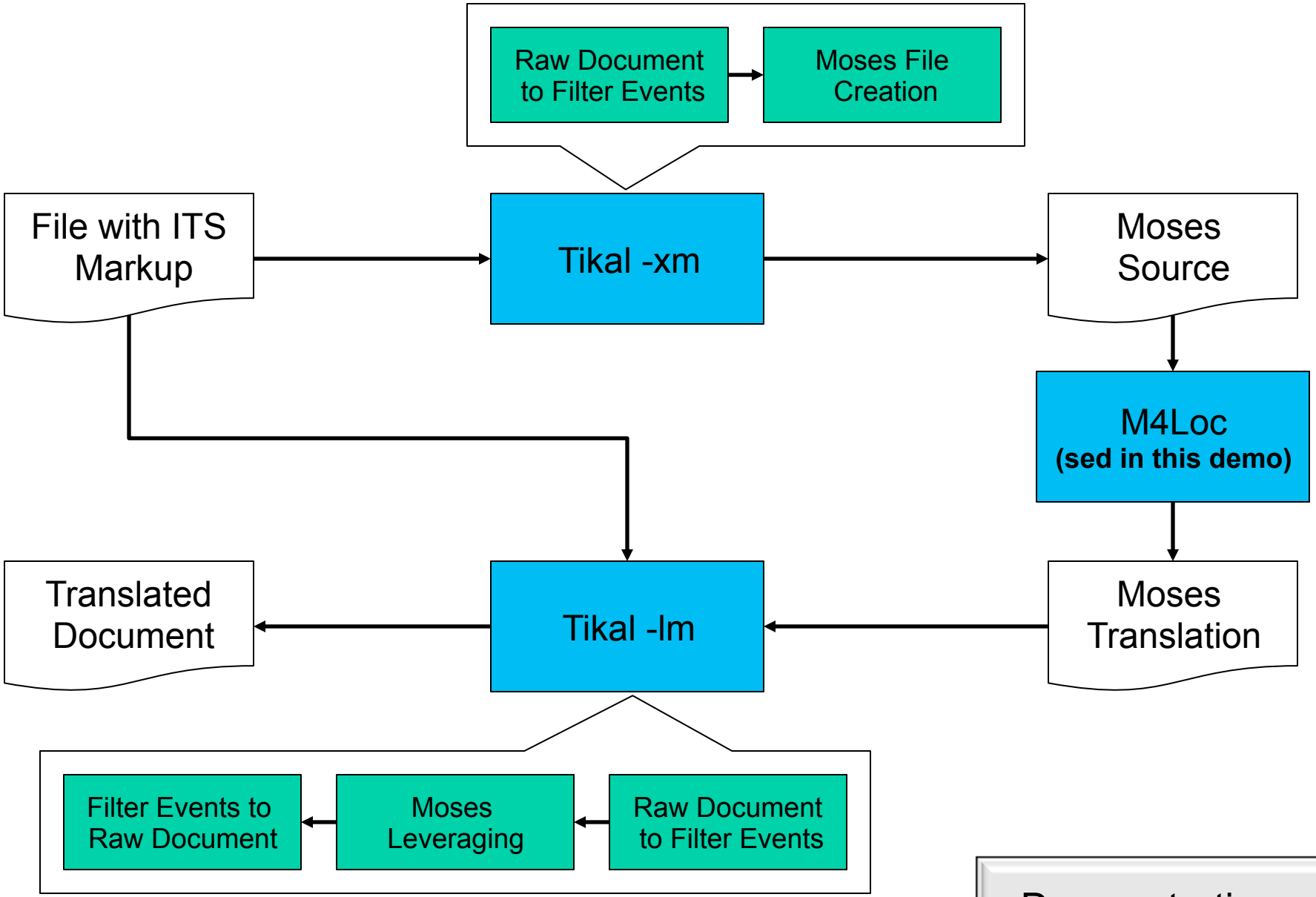
Benefits

- The ITS markup provides the key information that drives the extraction in both XML and HTML5.
- Information such as preserving white space can also be passed on to the extracted content and insure a better output.

Moses Translation (M4Loc)

- **Translate** - The non-translatable content is protected.
- **Locale Filter** - Only the parts in the scope of the locale filter are extracted, the others are treated as 'do not translate' content.
- **Element Within Text** - The information is used to decide what elements are extracted as in-line codes and sub-flows.
- **Preserve Space** - The information is passed on to the extracted text unit.
- **(Domain)** - The domain values are placed into a property that can be used to select an MT engine.

Moses Translation (M4Loc)



Quality Check

Description

- XML, HTML5 and XLIFF documents are read with ITS and loaded into CheckMate, a tool that performs various quality verifications.
- The XML and HTML5 documents are extracted based on their ITS properties, and their ITS metadata are mapped into the extracted content. The XLIFF document is also extracted and its ITS-equivalent metadata also mapped.
- The constraints defined with ITS are verified using checkMate.

Data Categories

- Translate
- Locale Filter
- Element Within Text
- Preserve Space
- Id Value
- Storage Size
- Allowed Characters

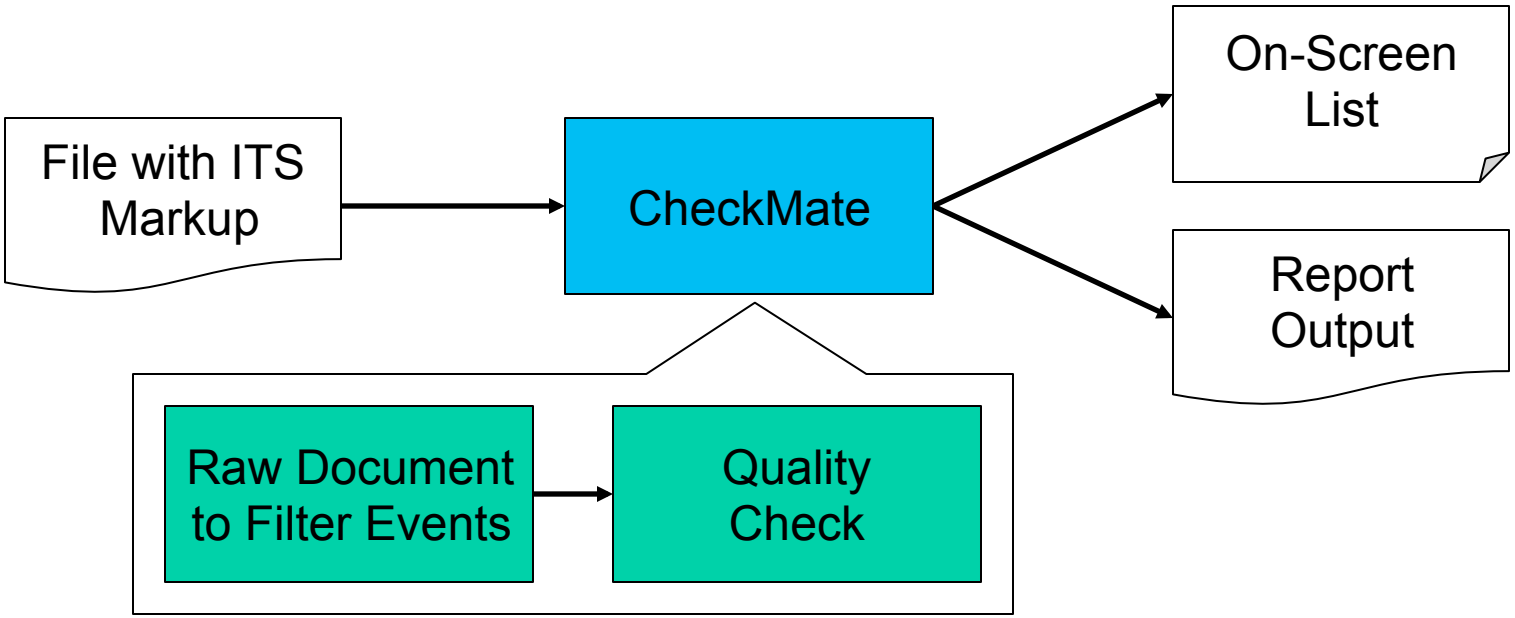
Benefits

- The ITS markup provides the key information that drives the extraction in both XML and HTML5.
- The set of ITS metadata carried in the files allows the three file formats to be handled the same way by the verification tool.

Quality Check

- **Translate -** The non-translatable content is protected.
- **Locale Filter -** Only the parts in the scope of the locale filter are extracted, the others are treated as 'do not translate' content.
- **Element Within Text -** The information is used to decide what elements are extracted as in-line codes and sub-flows.
- **Preserve Space -** The information is mapped to the preserveSpace field in the extracted text unit.
- **Id Value -** The ids are used to identify the entries with an issue.
- **Storage Size -** The content is verified against the storage size constraints.
- **Allowed Characters -** The content is verified against the pattern matching allowed characters.

Quality Check



Demonstration…

More Information

- Project wiki:
  http://www.opentag.com/okapi/wiki/
- Project source code:
  http://code.google.com/p/okapi/
- Continuous integration:
  https://okapi.ci.cloudbees.com/
- Maven repositories:
  http://repository-okapi.forge.cloudbees.com/release/
  http://repository-okapi.forge.cloudbees.com/snapshot/
- Developers mailing list:
  https://groups.google.com/group/okapi-devel/