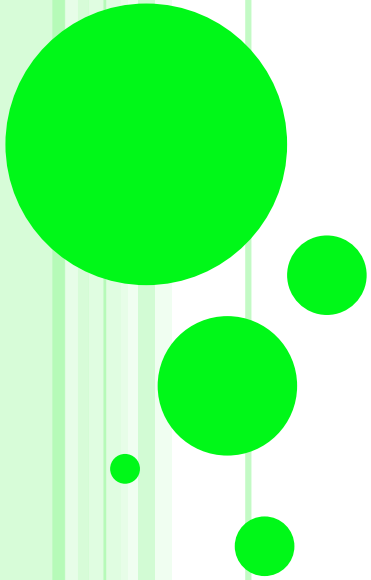
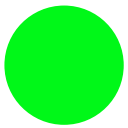


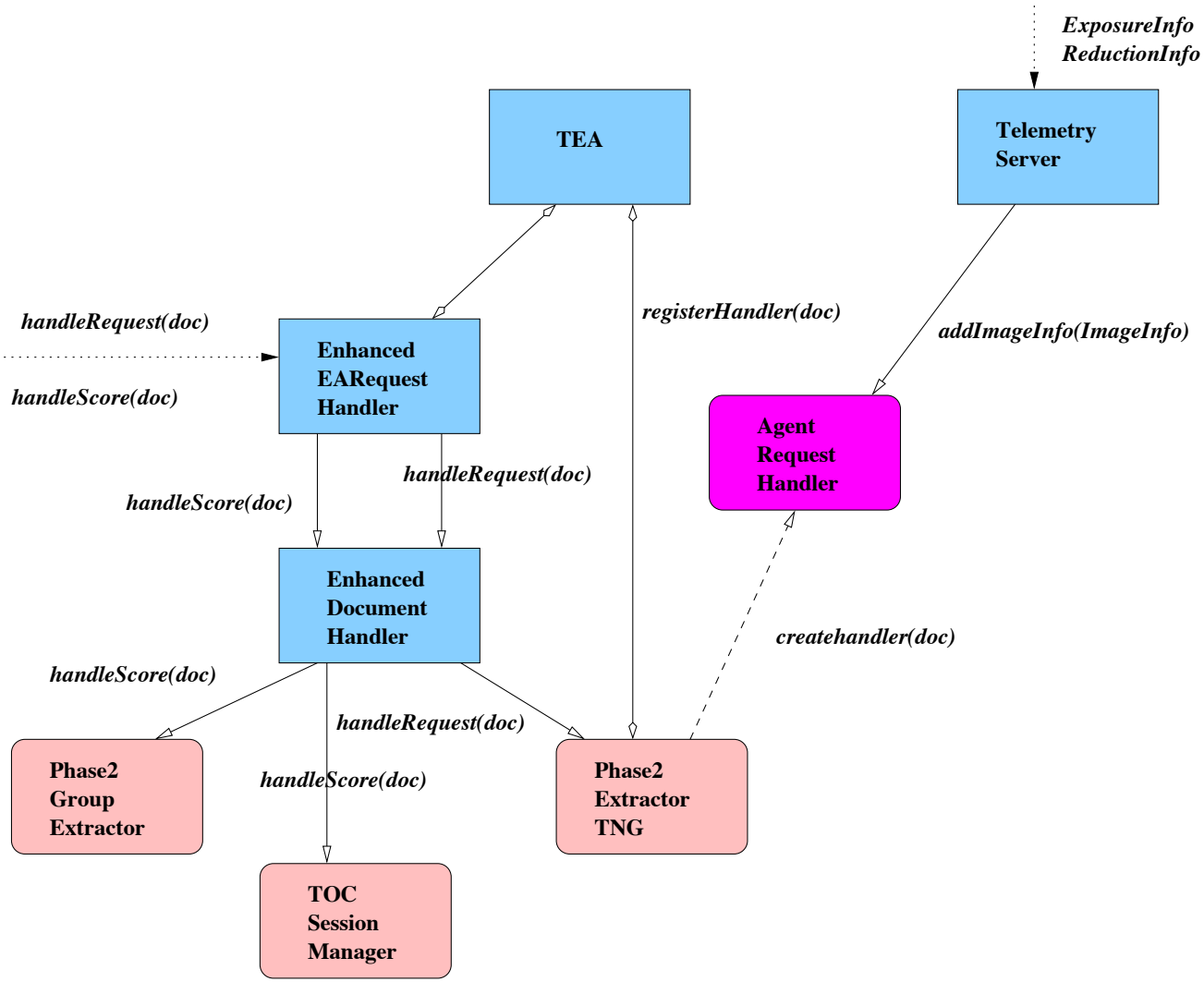
RTML and NSO Interfaces

Telescope Embedded Agent

- The TEA was written several years ago and has gone through several iterations.
- Originally used estarIO as comms layer.
- Now uses RTML requests from the NodeAgent.
- Requires potentially rapid telemetry from the RCS to make asynch callbacks to remote agents.
- At the time of writing there was no suitable pub-sub telemetry system.
- Existing ports/holes were used to make a fudged solution – CAMP.
- Code is in *~dev/src/estar/tea/* and is under svn or maybe rcs ?

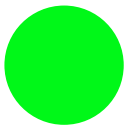


ARCHITECTURE



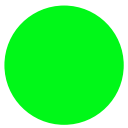
OPERATION

- Requests from the NA are passed to the *EmbeddedAgentRequestHandler* via calls to:-
  - handleScore(RtmlDoc) - scoring
  - handleRequest(RtmlDoc) – request to observe
- The observation, timing and constraint information is extracted and either a scoring decision or observation request may be made.
- TO requests are handled by
  - *TOCSessionManager*.
- Scheduled requests are handled by
  - Phase2ExtractorTNG.
- In the case of a scheduled request, an observation request is created and entered into the Phase2 ODB via the OSS. (rmi calls to Phase2Model etc).
- An *AgentRequestHandler* (ARQ) thread is started for each document currently in the Phase2 ODB to monitor for execution and make callbacks to the remote agent.
- Updates to the ARQ are received by the *TelemetryServer* in the form of:
  - ObservationInfo (a group has started)
  - ObservationCompletionInfo (a group has completed)
  - ExposureInfo (an image was taken)
  - ReductionInfo (a reduced image is ready)
- ARQ handlers are persisted over restarts by a simple serialization mechanism.



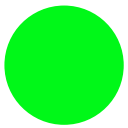
Problems

- The TEA is not very efficient – it was originally written to handle a few observation requests.
- The ARQs are separate threads – it would be better to have a single thread handle all the documents in round-robin fashion – much more scalable.
- The Telemetry mechanism is crude – there are modern feeds which could do this job anyway - OPS feed.
- The TEA keeps a local cache of proposals, groups and instrument-configs – this could be done nowadays via the Phase2 cache via rmi.
- RTML is structured like the old phase2 system.
- It is therefore neccessary to translate from groups of observations to sequences and a certain amount of deduction is needed.



NSO Interface

- This is an XML based interface.
- Phase 2 markup language (P2ML).
- Version (1) – based on the old phase2 with observations inside groups.
  - Most NSO observations still use this model.
  - They have to be translated into new sequences and constraint types.
- Version (2) – based on modern sequences.
  - Some NSO observations use this.
- All code is in the misc-systems repository and is in the default package.
  - V1 uses a SAX parser – it is evil. Implemented via *XMLParser* class.
  - V2 uses JDOM – it is great. Implemented via *P2mlParser* class



Operation

- Something on the NSO site generates XML documents. These are read by a client and sent via TCP to a server on *ltproxy*.
- Offline Control Relay (OCR) – cant recall where the name came from.
- Started using: */etc/init.d/ocr_init start*
- All the code is deployed to
  - */proxy/misc/misc_systems.jar*.
- Scripts are in
  - */proxy/bin/*
- Like the RTML interface – V1 requests need to be translated to sequences – a fair amount of deduction is needed.