Functional description of the Local Sky Model (LSM)

J.E. Noordam, R.J. Nijboer

(jnoordam@astron.nl)

Dwingeloo

Version 1.1: 8 July 2005

Verified:			
Name	Signature	Date	Rev.nr.
	o.p.v.	8th July 2005	0.9 0.9.5

Accepted:		
Work Package Manager	System Engineering Manager	Program Manager
J.E. Noordam .... *date:*	C.M. de Vos .... *date:*	J. Reitsma .... *date:*

Distribution list:

Group:	For Information:
ASTRON	dOList
O.M. Smirnov (OMS)	C.M. de Vos
R. J. Nijboer (RJN)	K. van der Schaaf (KvdS)
A. G. Willis (AGW)	H. Holties
R. Assendorp (RXA)	A.G. de Bruyn
T. A. Oosterloo (TAO)	M. A. van Haarlem
G. van Diepen (GVD)	J. D. Bregman (JDB)
W. N. Brouw (WNB)	
M. Brentjens (MAB)	
M. Mevius (MXM)	

Document revision:

Revision	Date	Section	Page(s)	Modification
0.5	2005-July-01	-	-	Creation
1.0	Version 1.1: 8 July 2005	??	??-??	Updated class description

Abstract

The Local Source Model (LSM) is a cornerstone of MeqTree data reduction for radio aperture synthesis. This functional description is the basis of its implementation.

1 Introduction

A Local Sky Model (LSM) is a temporary object, created for use with a specific uv-data set (MS), or group of such data-sets. It has three interfaces with the outside world (see fig 1):

1. Upon request, the LSM calculates values for the 4 Stokes *manifestations* (I,Q,U,V) of a given source or patch, for the cells of the specified domain(f,t). It does this by means of MeqTree subtrees, which are linked up with suitable *receptor* nodes of a MeqTree *User Forest*. The MeqTree system also allows for the updating of LSM source parameters after solving for them.
2. The LSM is also able to analyse the (residual) images made from the uv-data set(s), in order to update existing LSM sources, or to find new ones (see section 5).
3. The LSM will usually be connected with a Global Sky Model (GSM), which provides it with an initial subset of sources, and which will eventually be updated from the updated LSM (see section 6).

The Stokes manifestations are used by the User Forest to calculate the contribution of LSM sources and/or patches to predicted visibility values (see fig 6). These are used either to solve for parameters of the Measurement Equation (which include instrumental *and* source parameters) by means of selfcal, or to subtract sources from the uv-data. Thus, the internal LSM subtrees are part of a single M.E. The conversion to visibility values is done by the User Forest, and thus outside the scope of the LSM.

The LSM will usually deal with *intrinsic* source parameters, just like the GSM. The LSM observational window (i.e. the idealized station power beam) provides a *first-order* relation with apparent values. The User Forest deals with the station-dependent parts. However, it is also possible to use the LSM in 'apparent' mode (i.e. using apparent fluxes), as is the practice in most existing data reduction packages.

During the evolution of the LSM, MeqTrees have tended to play an ever increasing part. Whenever a problem arose, MeqTrees usually provided the answer, often with unexpected extra possibilities. We expect this process to continue in the first few iterations of LSM implementation. Much of this was glimpsed in an early stage by Oleg Smirnov[OMS2003].

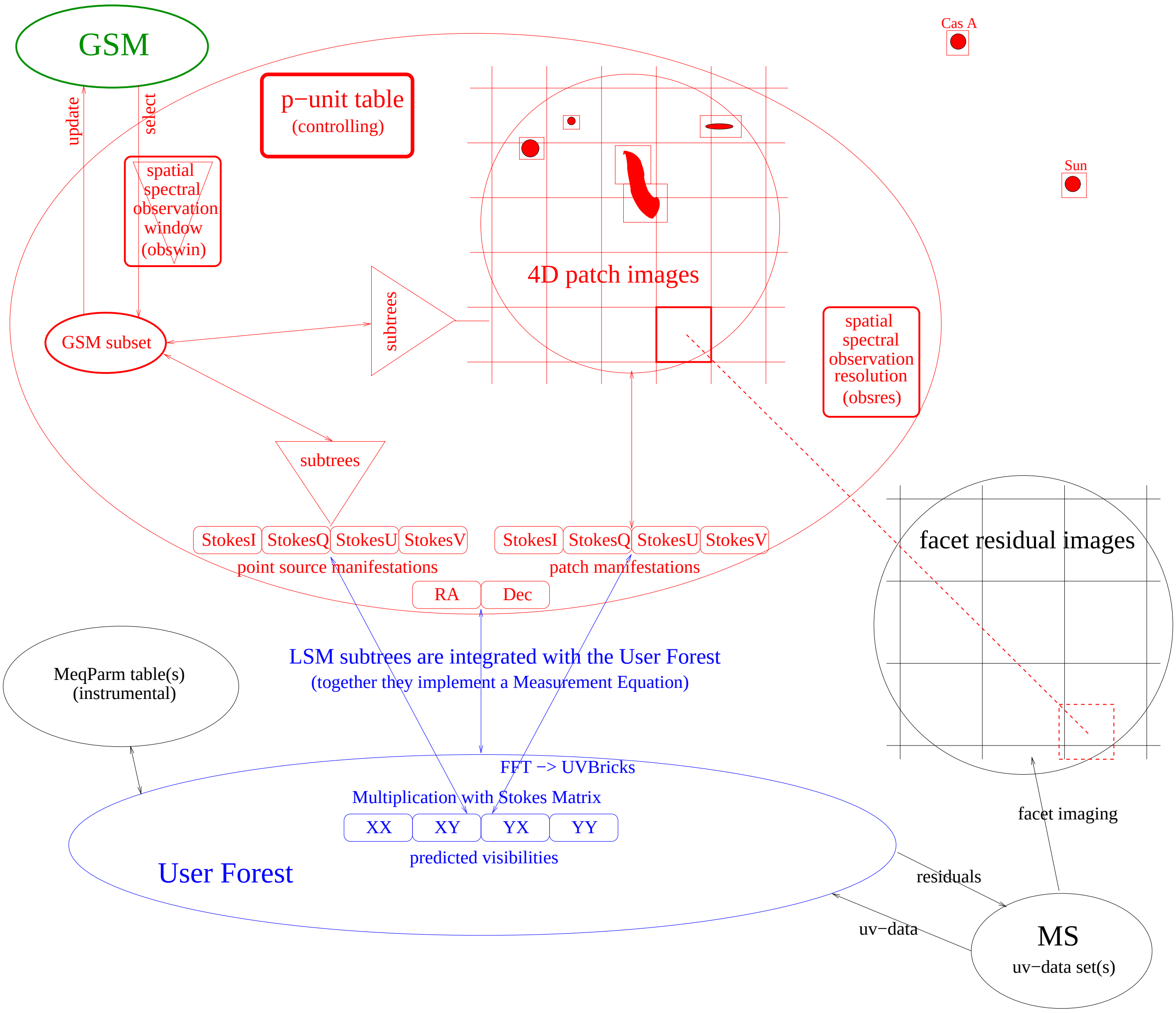

Figure 1: *Blob-diagram of the LSM (red) and its environment.*

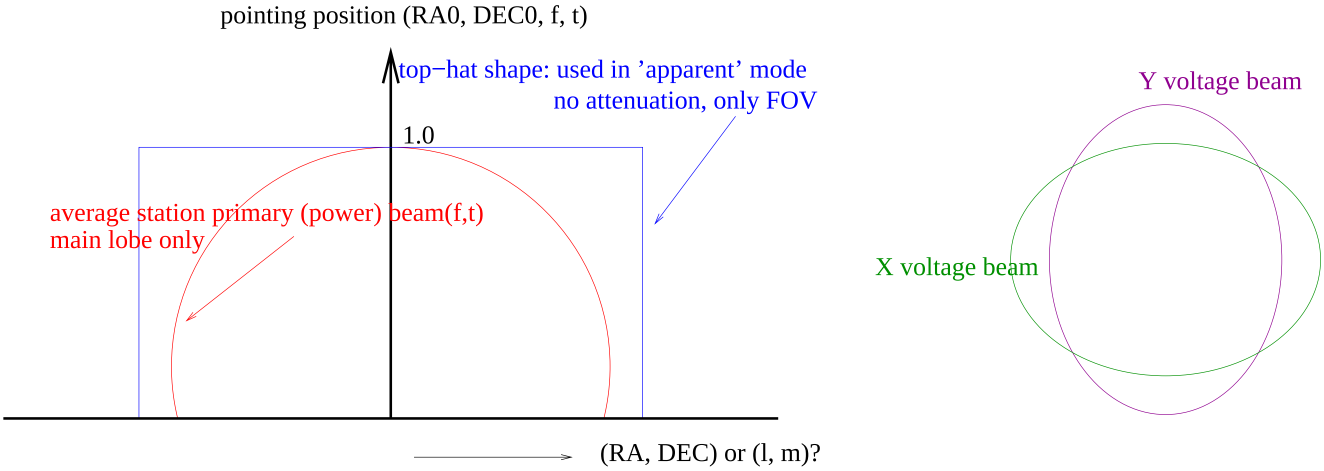

Figure 2: *The observational window (*OBSWIN*) is used to select LSM sources from the GSM, and as a first-order conversion between intrinsic and apparent fluxes. Since this is an idealized shape, it can be implemented as MeqTree subtree(s), with parameters. This opens the possibility for the* OBSWIN *to be time-variable as well as freq-variable, and for solving for its parameters. The* OBSWIN *is applied by inserting its subtree(s) into the various source prediction subtrees.*
Example: A WSRT telescope (station) has an equatorial mount, so that its two linear dipoles (X and Y) do not rotate w.r.t. the sky. Because of the size and orientation of a dipole, the shape of its voltage beam on the sky is slightly elongated (top right). Thus, the OBSWIN *shape for the StokesI (=XX+YY) power beam will be circularly symmetrical. The* OBSWIN *shape for StokesQ (=XX-YY) will have a clover-leaf pattern, multiplied with the StokesI beam. Since it is more difficult to say anything about the StokesU and StokesV beams, we will use the StokesI beam for the moment.*
Obviously, the OBSWIN *is most useful in those cases where the station primary beams are roughly similar. This will usually be the case for the main lobe, and perhaps the near sidelobes, but certainly not for the far sidelobes of LOFAR. Thus, the* OBSWIN *subtree should not be part of the prediction trees of sources far outside the FOV, like Cas A or the Sun.*

2 Components of the LSM

The LSM has a number of distinct components.

2.1 The observation window (OBSWIN)

See fig 2.The LSM OBSWIN is an idealized shape of the average station *power* primary beam. It can be seen as a *first-order* conversion between instrinsic and apparent fluxes.

- The OBSWIN is *essential* for selecting sources from the GSM. Therefore, the LSM should always have at least an OBSWIN size to indicate the FOV, even in 'apparent' mode.
- The OBSWIN is *useful* for estimating source parameters from residual images, because it substantially speeds up the convergence.

Obviously, the OBSWIN is most effective with station arrays with roughly similar primary beams. This will usually be the case for the main lobes, and even the near side-lobes. But certainly not for the far side-lobes of LOFAR stations, so the OBSWIN subtree should not be part of the prediction trees of bright sources outside the FOV, like Cas A or the Sun.

Since the OBSWIN is an idealized shape (see fig 2), it can be defined by means of a parametrized analytic function. Therefore, it can be implemented as MeqTree subtree(s) with MeqParm parameters. This is very powerful, because not only does this allow the OBSWIN to be time-variable as well as freq-variable, but it also opens the possibility of solving for the OBSWIN parameters.

NB: The OBSWIN might be defined in terms of (RA, DEC), which makes it easy to calculate its attenuation of a particuar source. However, this causes problems around the pole, so it could be better to define it in (l,m), referred to the pointing centre (RA(t), Dec(t)) of the observation....??

Usually, the necessary information (TDL subtree definition from the MS?) to define the OBSWIN will be supplied when the LSM is created, using information from the MS.

2.2 The observation resolution (OBSRES)

The LSM OBSRES (spatial and spectral) is similar to the OBSWIN, in the sense that it is an idealized quantity, defined with information from the MS. It may be used for quick-and-dirty CLEANing of residual images, in the way of the minor cycle of Clark CLEAN. It may also be used to determine the size of the uv-plane in a MeqUVBrick (provided we decide that the FFT is part of job of the LSM, which is probably not the case...).

The OBSRES could be stored as meta-data in the GSM, to indicate the limitations of the 'intrinsic' source parameters. *This opens a largish can of worms* (see also section 6).

2.3 A subset of the GSM (GSM-subset)

Using the OBSWIN, a subset of sources relevant for this particular observation is extracted from the Global Sky Model (GSM). See also section 6. Note that this subset will presumably have the same general structure as the GSM. In any case, it has the following parts (see fig 2):

- The **LSM source table**, with an entry for each available 'source'. This can be either a parametrized source component, or a 4D image of a 'special' source that cannot easily modelled with parametrized components. Each entry contains references to 6 subtrees, one for each standard manifestation (I,Q,U,V,RA,Dec).
- A **MeqParm table** that contains the parameter values for parametrized components. It also contains any solvable parameters for the special sources in 4D images. Note that, for each parameter, there may be more than one funklet (set of base function coefficients) available, for different validity domains(f,t).
- A collection of **4D GSM Images** (RA, DEC, freq, stokes) of 'special' sources that cannot be easily represented by parametrized components, e.g. very extended and irregular shapes. They are called 'GSM Images' to distinguish them from other kinds of images used in the LSM. In practice, GSM Images will often consist of CLEAN components. GSM Images may be associated with parameters in the MeqParm table.
- A set of 'unqualified' **templates** (probably in the form of TDL scripts) for the various types of **MeqTree subtrees** that are relevant for this subset. These are used to generate 6 actual subtrees for each source.

2.4 A collection (table?) of source patches

A patch is a gridded 4D image (RA, REC, freq, stokes) of a rectangular piece of the sky, which may contain zero or more sources from the *source-list* in the GSM subset mentioned above. It can also contain (part of) a very extended source that is stored as a 4D image in the GSM.

A patch is implemented as 4 MeqLSMPatch nodes, which can be inserted in a LSM subtree. Patches are discussed in some detail in section 4.

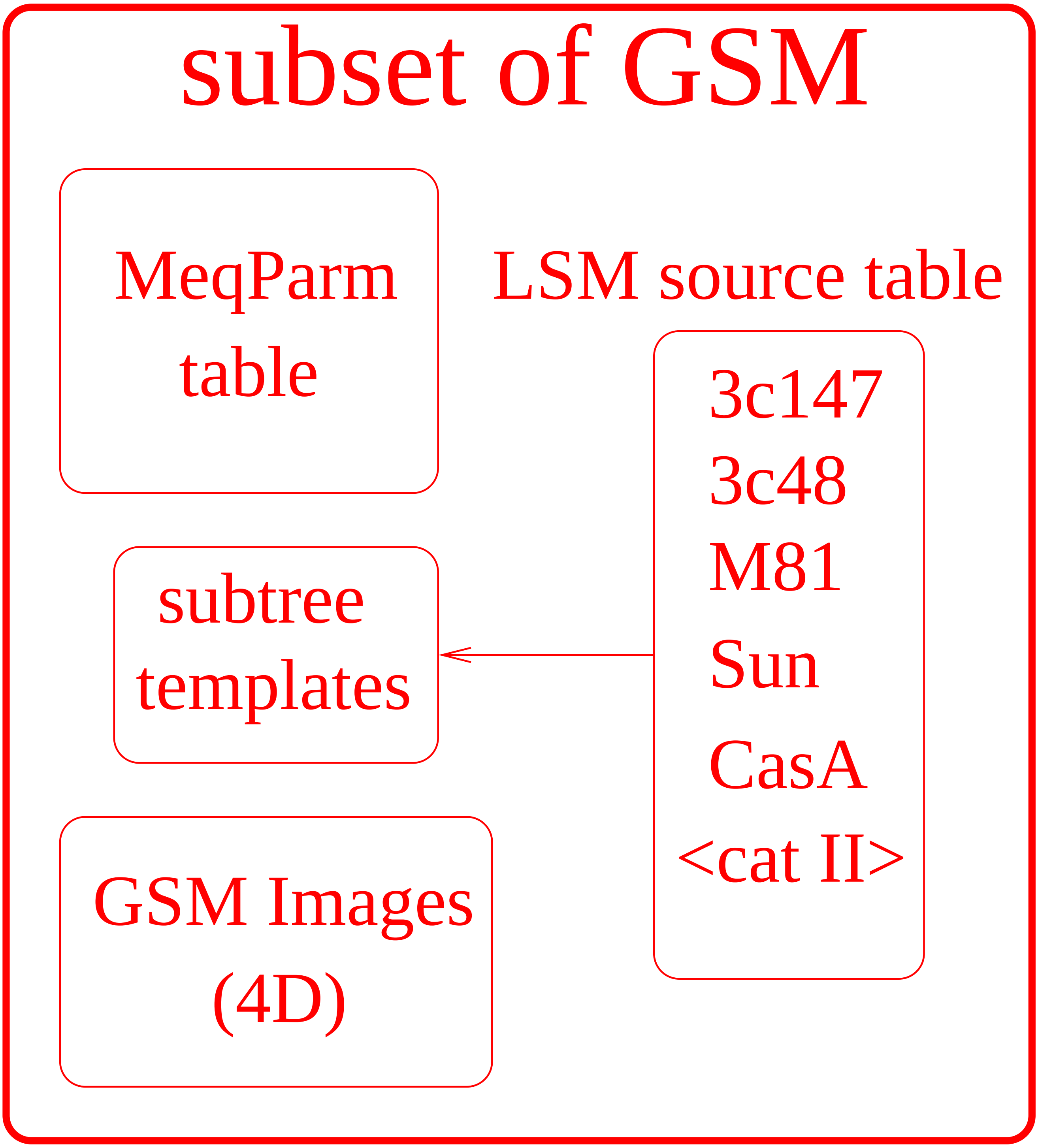

Figure 3: *Structure of the GSM subset. Presumabyl, the GSM will have the same general structure, but it may be implemented differently.*

In fig 1, the patches in the list are projected on the sky. The circle represents the FOV (OBSWIN), which may be tiled by patches that contain groups of Cat II sources (point and extended). Extended Cat I sources usually have their own patch. Patches may overlap, but a particular source may only occur in one patch, of course.

2.5 A table of prediction units (p-units)

The p-unit table is the **controlling component** of the LSM. Each entry represents either a point source, or a patch. It has the following columns:

- **P-unit name:** For instance '3c147', or 'patch#23'. Note that patches for extended sources may also be named after this source (e.g. M81). The name is used in the MeqTree node qualifiers, both inside the LSM, as in the User Forest.
- **P-type:** At this moment, this can either be 'point' (point source) or 'patch'....?
- **P-unit source list:** A list of one or more source names (from the LSM source table) that contribute to this P-unit. Note that a patch does not know which LSM sources it contains, but this information is avaliable here.
- **Cat I switch (T/F):** Indicates whether a p-unit is a Cat I source, i.e. whether it is used for selfcal and *individual* subtraction.
- **Apparent brightness (Jy):** This is needed to determine the peeling order of the Cat I sources. Presumably, the p-unit table will be ordered in reverse order of apparent brightness[1].
- 6 columns (StokesI,Q,U,V and RA,Dec) that contain the **root node names** of the corresponding subtrees. These are used as receptor nodes to link up with the User Forest, and for other tasks.

3 Use of the LSM in practice

Just to get a feel for the use of a LSM, here is a very general procedure:

- Select an uv-data set (MS):
  - Get its spatio-spectral observation window (OBSWIN), possibly in the form of a TDL script stored in the MS.
  - Get its observation resolution (OBSRES).

[1]What is meant is the apparent brightness at short baselines. Long baseline uv-data samples for which the source is no longer visible will be ignored in the selfcal solution.

- Create a LSM for this particilar observation, using the OBSWIN and OBSRES:
  - Use the OBSWIN to select a suitable subset of sources from the GSM, and put these in the LSM source table.
  - Copy the relevant part of the GSM MeqParm table, and the relevant 4D GSM Images.
  - Define the various patches (extended sources and FOV tiles), and identify their contributing sources in the LSM source table.
  - Generate the p-unit table of point sources and patches.
  - Generate subtree(s) for the OBSWIN, to be inserted in source prediction subtrees.
  - Generate the various subtrees for each source/patch, and put their root node names into the p-unit table. Subtrees for sources in the main lobe will contain the OBSWIN subtree.
- Select an operation to be performed on the MS (i.e. a TDL script, usually in Python).
- Generate the User Forest from the selected TDL script. If it implements a peeling operation, it will use the LSM p-unit table to generate *peeling-units* in reverse order of apparent brightness. Their p-type determines the kind of *receptor* nodes that are needed (e.g. MeqUVInterpolate or not) to connect to the 4 Stokes manifestation nodes for this p-unit.
- Attach the MS to the User Forest, and execute it. If the forest solves for LSM source (or patch!) parameters (e.g. by selfcal), these are updated in the LSM.
- If the User Forest produces residual uv-data, do the following:
  - Create facet residual images. Then invoke the LSM to analyse the facets, using the p-unit table to know where to look. In this way, existing (Cat II) sources may be updated. In addition, new (Cat III) sources may be identified, and added to the LSM.
- After the last iteration, and if the LSM has been modified:
  - Update the LSM source parameters from the patches.
  - Update the GSM from the LSM subset

With experience, we will probably think of various other ways of using an LSM.

3.1 Special case: Use of the LSM without GSM

It should be possible to use an LSM without the benefit of a GSM. The main difference will be the generation of an initial LSM source model. Sources could be specified by hand (like with 3C343), or generated by shallow CLEANing of a dirty image. The latter is equivalent to starting halfway the above recipe. The fainter sources would emerge automatically in later iterations.

Even without a GSM, an OBSWIN is still needed for defining tile patches. But one might contemplate to ignore the shape of the OBSWIN, and work in apparent mode (see below).

3.2 Special case: Use of the LSM in 'apparent mode'

If operating without a GSM, iit should also be possible to work in 'apparent' mode, i.e. with an OBSWIN that is set to unity everywhere (see fig 2). In fact, this is the mode in which most reduction packages work at the moment.

3.3 Disk storage and retrieval of an LSM

With or without a GSM, it may often be useful to be able to store a LSM on disk, and to retrieve it later for further processing.

4 Calculating IQUV manifestations for a p-unit

A p-unit is associated with 6 MeqTree subtrees, whose root nodes have the following standard[2] names:

standard node name	remarks
StokesI[q=<qual>]	total intensity (Jy)
StokesQ[q=<qual>]	linear pol (Jy)
StokesU[q=<qual>]	linear pol (Jy)
StokesV[q=<qual>]	circular pol (Jy)
RA[q=<qual>]	Right Ascension (rad)
Dec[q=<qual>]	Declination (rad)

The nodename qualifier <qual> is an indication of the *p-unit name*, which may be an actual source name (e.g. 3c343, or Sun), or a patch indication (e.g. patch#23). The TDL script that generates the User Forest knows from the p-unit table whether to expect a MeqUVBrick or not, and connects the 6 standard nodes as children to suitable receptor nodes. The LSM subtrees will have been generated from templates when the LSM was created, so the node names in the p-unit table are sufficient reference.

[2] The notion of standard receptor node names was essential for earlier tree generation schemes, where canned trees link up automatically with a given LSM. Standard names are no longer necessary with the new TDL generation scheme, but they are retained for clarity, and because they might prove useful in the future.

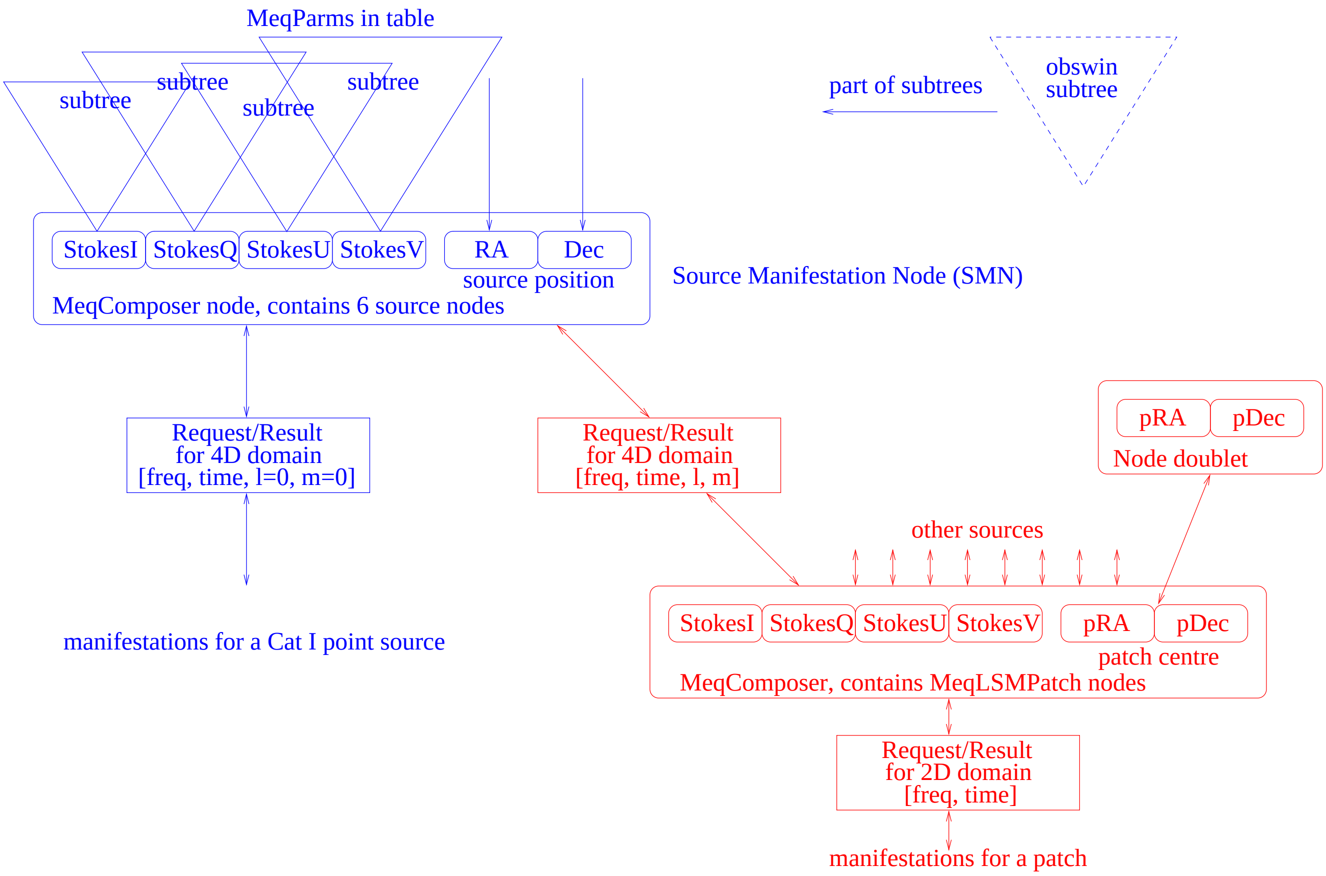

Figure 4: *A Source Manifestation Node (SMN) contains the root nodes of 6 subtrees (StokesI,Q,U,V, RA, Dec) that implement the particular parametrization of a parametrized source. The 'leaves' of these subtrees are MeqParms, which may be solved for by the User Forest, using selfcal.*
Bright (Cat I) point sources are connected directly to the User Forest. Extended sources, and groups of fainter (Cat II) sources, are predicted by means of patches, which are implemented as MeqLSMPatch nodes that have one or more SMN children. GSM images are also predicted by means of patches, in a way that is not indicated here.
Note that the OBSWIN *may be applied to point sources and patches, by integrating its subtree into the various prediction subtrees. The root nodes of* OBSWIN *subtrees take requests with 4D domains (freq, time,l, m), just like the SMN's.*

The 6 (I,Q,U,V, RA, Dec) manifestations of a parametrized source component are calculated from its particular parametrization by its 6 subtrees (see fig 4). The request (and thus the result) is for a 4D domain (freq,time,l,m)[3]. The spatial coordinate (l,m) grid is related to the source position, so the source (RA,Dec) coincides with (l=0,m=0). The shape parameters of extended sources are given in terms of these (l,m) coordinates (see fig **??**). For convenience, the 6 nodes may be bundled into a **Source Manifestation Node (SMN)**, which is a MeqComposer node containing the 6 manifestation nodes. The root nodes of obswin subtrees take requests with 4D domains (freq, time, l, m).....?

Note that the OBSWIN *should be applied by integrating a suitable subtree into the various source prediction subtrees. Thus, the patches always contain apparent (to first order) fluxes.*

If the source is a bright (Cat I) *point source*, the 6 manifestations (or their SMN) may now be linked to the User Forest receptor nodes for further use. Its source parameters, residing in the LSM MeqParm table, may be solved for in the usual way.

If the source is extended, or if it is a Cat II source, its manifestations are transferred to a patch. This case is treated in the next section.

If the source is a GSM Image, it must be mapped onto one or more patches. This is not a trivial matter, with (l,m) grids etc. We tackle it after we have some experience with parametrized components.

4.1 MeqLSMPatches

A patch is a very important kind of LSM p-unit. It could be implemented as 4 **MeqLSMPatch** nodes, which may be bundled together for convenience, like the SMN. A MeqLSMPatch node contains an internal (l,m) grid centered on the patch centre (pRA, pDec). Upon receiving a request, the MeqLSMPatch node adds 2 dimensions (l,m) to its input domain and issues it to its children, which will include one or more SMN's. Since each input source is defined in its own (l,m) coordinate system, they must be converted to the patch grid.

A patch will usually either contain a single (bright, Cat I) extended source, or all the Cat II sources in a 'tile' of the FOV. The only difference is that one will probably not try to solve for source parameters in the latter case (although one could!).

A patch can be any size. However, if it is to be used for peeling it has to be small enough to be able to accurately interpolate the resulting UVBrick. In that case, it also has to be padded with zeroes (see below). Three kinds of patches are envisaged:

- Tile patches, which cover a small part of the tiled FOV. Their (l,m) grid is defined when they are created, and is added to the domain of a request. A MeqLSMPatch node is defined by the positions

[3]If a request is made for a 2D domain(freq,time), the node should assume (l=0,m=0).

and size of its child sources. This solves any problems with tiling the FOV with patches with differen (l,m) grids.

- Patches that contain a single (bright, Cat I) extended source. Since we do not want a patch to be larger than necessary, we need a mechanism to let the SMN decide for how many (l,m) points to calculate values. This begs the question about the size of the (l,m) grid in the request issued to the SMN.
- Patches that contain (a part of) a GSM Image. These will be tackled in a later stage.

A patch may be transformed by FFT into gridded uv-data in a UVBrick by the User Forest. This is the point where it may be decided to pad the patch with zeroes on the outside, in order to create a finer (u,v) grid in the UVBrick.

In peeling mode, the phase-centre of the uv-data will coincide with the p-unit centre position, irrespective whether it is a patch ot a point source. However, this will not be the case if the p-unit is 'taken into account' in selfcal, in order to reduce effects of peeling 'contamination'[4]. One way to deal with this is to convert the (l,m) grid of the p-unit into one that is referred to the phase centre. Alternatively, its UVBrick visibilities must be multiplied with a phase term (KJones) before adding their contribution to the predicted visibilities. Obviously, they must first be corrupted with instrumental errors in the direction of the p-unit. The same goes for simultaneous solutions for multiple sources. The only difference is that we now solve for more than just the instrumental errors in the direction of the peeling source.

5 Updating source parameters

LSM source parameters may be updated during processing, and new LSM sources may be created. The updated LSM may eventually be used to update the GSM.

5.1 By means of selfcal

If a source is bright enough, it can be used to estimate parameters by means of selfcal, i.e. by comparing predicted with measured uv-data. This is often done in the so-called 'peeling' mode, in which the sources are treated one by one, in reverse order of brightness. We may solve for source parameters, or instrumental parameters in in the source direction. As usual, two cases may be distinguished:

- **Point sources:** LSM point-source subtrees are the 'twigs' of the user forest, which is now directly

[4]This is the contamination from sources away from the phase centre, i.e. not the current peeling source, on the measured visibilities.

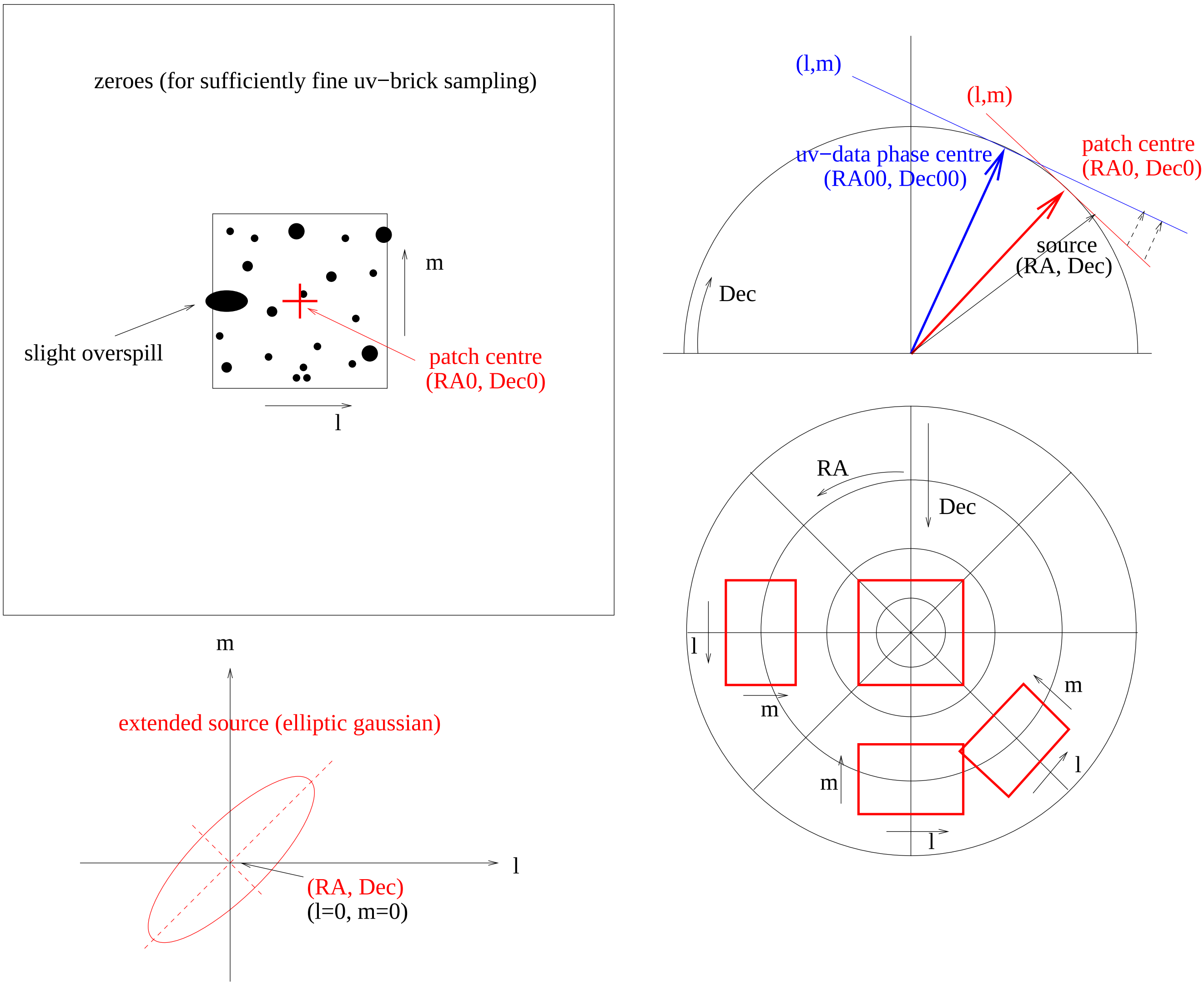

Figure 5: *In principle, a patch can have any size. However, if it is used for peeling, its size will be limited, and the the outside will be padded with zeroes in order to get sufficiently fine sampling in the uvbrick after FFT.*
At the right, there are some visual aids to understanding the (l,m) discussions in this document.
An extended source is defined in a (l,m) coordinate system centered on its own position (RA, Dec) at (l=0,m=0): $exp^{-(a.l^2+b.m^2+c.l.m)}$

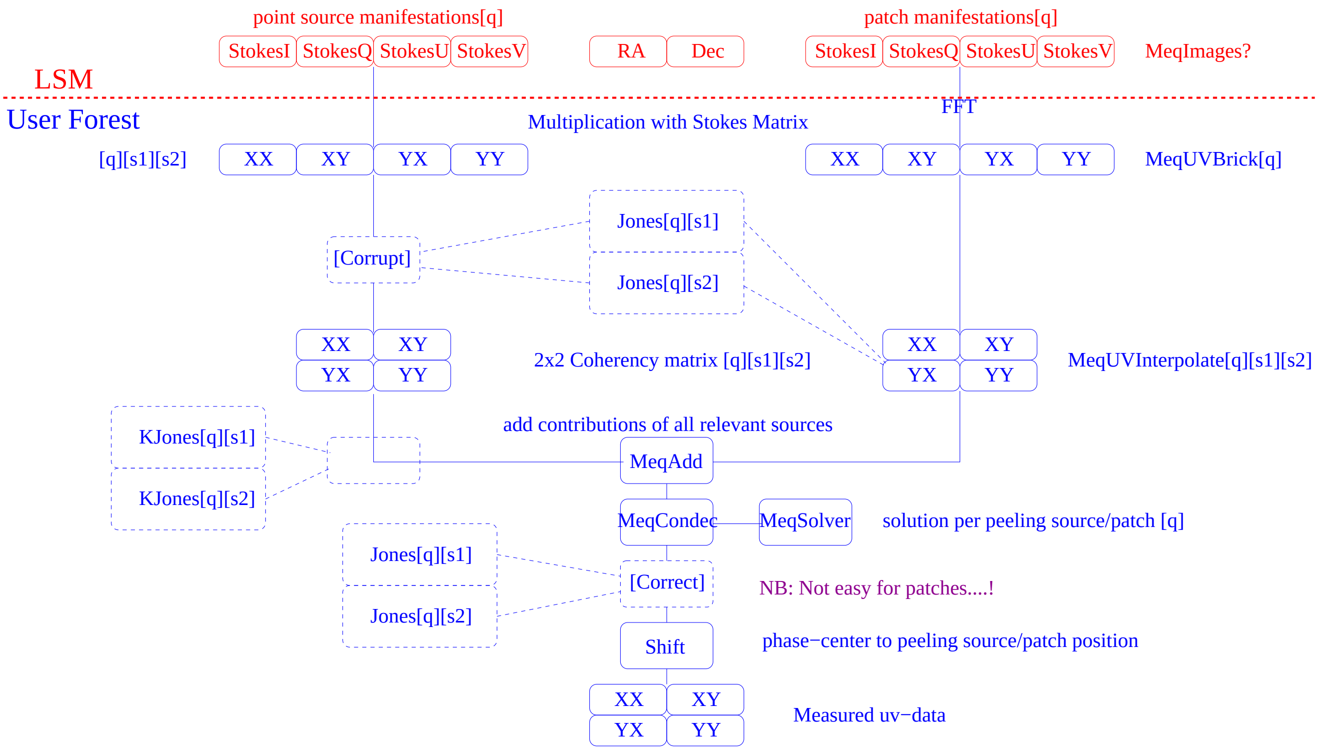

Figure 6: *Predicting source visibilities from the LSM Stokes manifestations (red), for the two cases of point sources and patches. Note that instrumental effects (Jones matrices) may be applied on either side of the solver, depending on the application.*
MeqLSMPatch nodes (patches) may be converted by FFT into 4 MeqUVBrick nodes, which contain the 3D gridded visibility function (u, v, freq) of a patch, for the entire observation (all baselines, all times and frequencies). Note that the FFT is somewhat unusual, because we require that (u,v) are measured in meters, rather than wavelengths. Visibilities for individual baselines [s1][s2], for a specific domain(f,t), are obtained by interpolating the UVBrick. Instrumental effects, even the ones that vary over the patch, may be applied by adding terms to the interpolation function (see Bhatnagar et all). For efficiency, we require that the number of frequency planes in the UVBrick is equal to the nr of domain cells in the frequency direction. If this is not the case, the UVBrick is recalculated. Thus, it pays to keep the frequency cells constant for as long as possible.
All this is shown here to illustrate that the LSM subtrees are integrated with the User Forest. Together they implement a Measurement Equation, which contains both instrumental and source parameters. ***However, calculating visibilities it is emphatically NOT part of the LSM.***

connected to the source parameter leaves. Therefore, point source parameters can be solved for in any combination with instrumental parameters.

- **Patches:** are implemented as MeqLSMPatch nodes, which have source subtrees as children. Therefore, it is possible to solve for the parameters of the input sources in the normal way. However, this is probably only practical for those cases where a patch represents a single bright extended parametrized source, like an elliptic gaussian (see fig 5). Nevertheless, patches that contain many fainter (Cat II) sources, or part of a GSM Image, could in principle be associated with a small number of 'overall' parameters like patch position or flux level, which could then be solved for. This has a low priority.

5.2 From residual images

Since selfcal is expensive, and most sources are too faint anyway, most sources are updated by inspecting residual images. Usually, some (or all) LSM sources will be subtracted from the uv-data. The uv-residuals are used to make residual images in the form of one or more *facets*[5] that tile the FOV (see fig 1).

The facet inspection process is controlled by the p-unit table in the LSM. Since facets and patches are completely independent, and thus may have different sizes and positions, a single p-unit may need to inspect more than one facet. *Moreover, there will be some issues with different (l,m) grids between p-units and facets. Inspection* could have the following stages:

1. Firstly, we may inspect the 4D facet residual images at the positions (l,m) where LSM sources are known to have been subtracted. Any deviation from zero (i.e. any hole or bump greater than the noise) is used to update the corresponding p-unit. As usual, two cases must be distinguished:
   - **Point sources:** The update corrections are applied directly to the source parameters in the MeqParm table. This is done by solving for these parameters, using their LSM subtrees, and the deviation from zero as the 'driving value'.
   - **Patches:** The source update corrections are applied at the corresponding position (l,m) in the relevant patch(es), but not yet to the underlying MeqParms. The patches are then used as input for the next *major cycle*[6] of the processing. At the very end of the reduction, the updated patch values may be used to update the MeqParms of those parametrized sources that contributed to this particular patch. Again, this is done by solving, using their LSM subtrees,

[5] Facets used to be necessary to make wide-field images with 2D arrays like the VLA at low frequencies. This is no longer necessary with the w-projection technique (Cornwell 2002), which can make a wide-field map in one piece. However, facets will still be needed for LOFAR, because the residual ionospheric phase errors over a facet should not be greater than about one radian. Fortunately, the size of a facet may be greatly increased, thus minimising the total number of facets, by correcting the uv-data for the linear term (i.e. phase gradient(f,t)) of the residual ionosphere before imaging. Facets should be made on the same sky-grid, so that they can be easily combined into larger images.

[6] A major cycle is an entire sequence of uv-data selfcal and source update via residual images. More than one major cycle will probably be needed to get the best dynamic range.

and the patch values as driving values. **NB: Here is a deviation from the MeqTree paradigm, since a MeqLSMPatch does not always interrogate its children when it gets a request. This may be address with a 'cache-switch'....**

2. Secondly, we may look for new parametrized LSM sources in the facet images, i.e. we try to identify Cat III sources and make them into Cat II sources. For each p-unit (patch), only the corresponding area in the facet image(s) is searched, of course. Such new LSM sources are created by generating source subtrees of a suitable type for them, and attaching these as children to the p-unit.

3. Thirdly, very extended residual flux may be stored, after deconvolution, in new 'GSM Images'. They are then assigned to one or more patches. This will be tackled in a later stage.

Since residual images are convolved with the PSF, and p-units are not, facet inspection will necessarily involve some sort of deconvolution. This is where the OBSRES, which can be seen as an approximation of the main lobe of the PSF, can play a useful role in shallow (inexpensive) CLEANing. However, it may be necessary to use the full PSF in CLEANing very extended Cat III structure. Either way, we may take into account that residual ionosphere errors causes the LOFAR PSF to degrade benignly, and in a known way, towards the facet edge.

6 Interaction with the GSM

The Global Sky Model (GSM) will probably have the same general structure as the LSM subset discussed above (see sect 2), but it may be implemented in a different way. Initially, there might even be different GSM's for different instruments (WSRT, GMRT, LOFAR), implemented differently.

The GSM will be different from existing astronomical catalogs in two major respects:

1. The source parameters will be stored as *funklets*, i.e. arrays of coefficients of suitable base-functions in time and frequency (e.g. polynomials). Constant values are funklets with a single coefficient, of course. A source parameter may have more than one funklet, each with its own validity domain(f,t).

2. Sources are associated with MeqTree *subtrees*, which represent the mathematical relation between the 6 source manifestations (I,Q,U,V,RA,Dec), and the source parametrization (and/or images). The parametrization can be anything, from NEWSTAR parameters to pixons or shapelets. The subtree definitions will probably be stored as unqualified templates of various subtree types, probably in the form of TDL scripts. When needed, these templates are converted into actual subtrees, in which the node names contain the source name as a qualifier.

The extra sophistication is very powerful, but has some obvious implementation (and maintenance) issues. These will not be discussed here. There are two main operations on the GSM:

- **Obtaining a GSM subset:** Use the LSM observation window (OBSWIN). Get all sources within the main lobe, and the bright sources that come in via the sidelobes.
- **Updating the GSM:** Just replace the source information in the GSM with the updated information in the LSM.

It is unclear what to do with the LSM observation resolution (OBSRES). It clearly is a property of an individual source, so it should not be stored as a GSM-wide quantity. But whereas it is always possible to predict a source manifestation for a wider PSF than the one for which it was derived, the reverse is much more problematic.

Finally, it seems logical to make any GSM compatible with the **Virtual Obervatory**, whatever that means in practice. Probably that we should take the initiative in defining as set of Universal Contents Descriptors (UCD) for our field, and to start using these consistently. At the very least, VO tools may be used to perform operations on the GSM, and possibly even the LSM.

7 Visualization of the LSM

Obviously, there should be various windows on the LSM. Ideally, these should be accessible via the MeqTree browser.

- Schematic overview of the relative position of all patches (like in fig 1), with some indication of the sources they contain.
- Same as above, with all point sources indicated in a separate color.
- Relative position of patches and facets (not strictly LSM).
- Rendering of individual patches, possibly with the relevant point sources superimposed in a different color. Such images should be selectable from the schematic overview of all patches.
- etc. etc

Note that the parts of the LSM that are implemented as MeqTrees may be visualized with the standard MeqTree node visualization tools. In addition, various built-in views of the LSM may be defined using the MeqDataCollect mechanism.

8 Conclusions

We appear to have a nicely consistent description of a very powerful LSM.

Obviously, its full implementation will be a lot of work. Many elements exist in an early form, but must be refined later. Other parts (e.g. sources stored as 4D images) we do not need yet. A basic LSM is needed before september, to do Abell963. After that, we must farm out the refinement of specific nodes to a 'cloud' of specialists.

GSM...

References

[OMS2003] Smirnov O.M., *The Global Source Model* (2003)

Contents