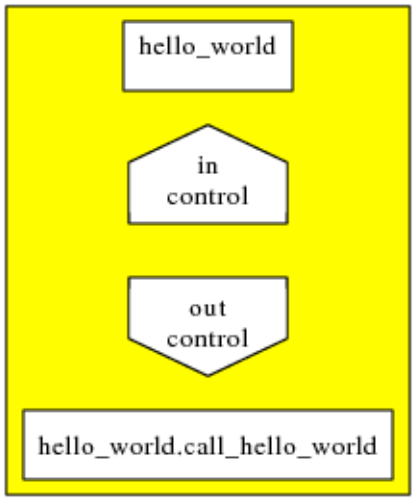
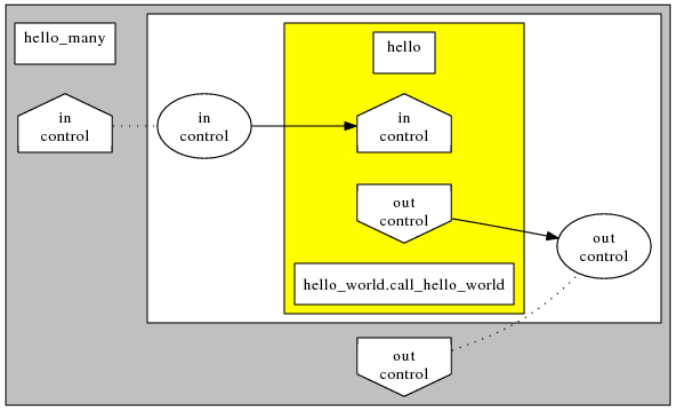
GPI-Space – Hello world

```
<defun name="hello_world">

  <in name="in" type="control"/>
  <out name="out" type="control"/>

  <module name="hello_world" function="out call_hello_world (in)">

    <cinclude href="hello_world.hpp"/>
    <link href="lib/hello2.o"/>

    <code><![CDATA[
      impl_hello_world();
      return control();
    ]]></code>
  </module>

</defun>
```



- function, port, types, external module call, build system

GPI-Space – Hello world, parall

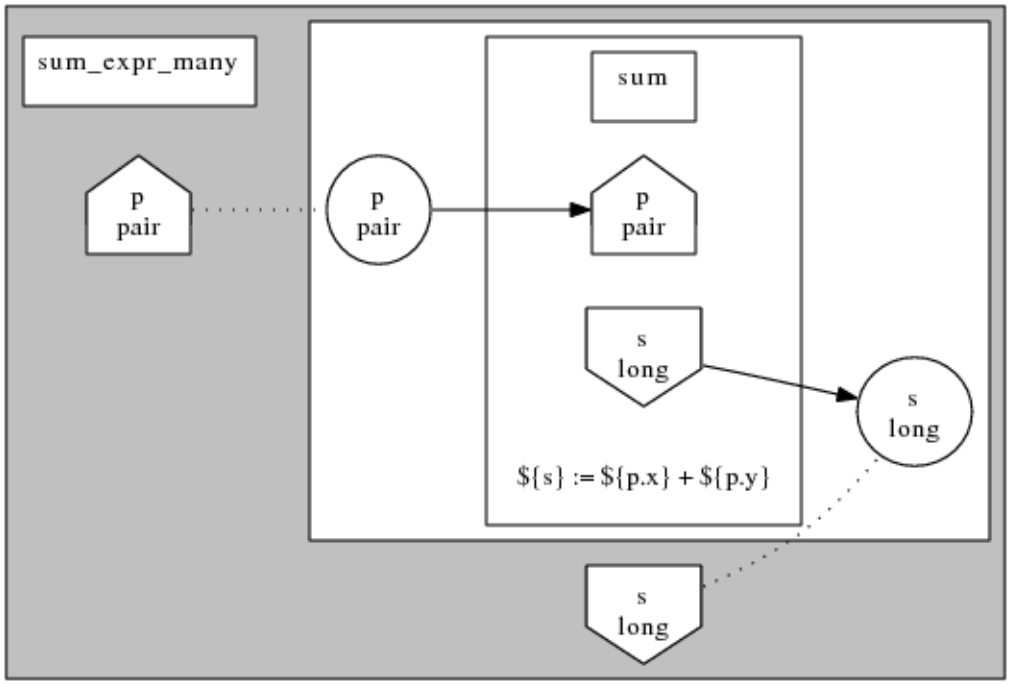



```
<defun name="hello_many">

  <in name="in" type="control" place="in"/>
  <out name="out" type="control" place="out"/>

  <net>

    <place name="in" type="control"/>
    <place name="out" type="control"/>

    <transition name="hello">
      <include-function href="hello_world.xpnet"/>
      <connect-in port="in" place="in"/>
      <connect-out port="out" place="out"/>
    </transition>

  </net>

</defun>
```

- subnet, connection, re-use of existing net, parallel execution, put

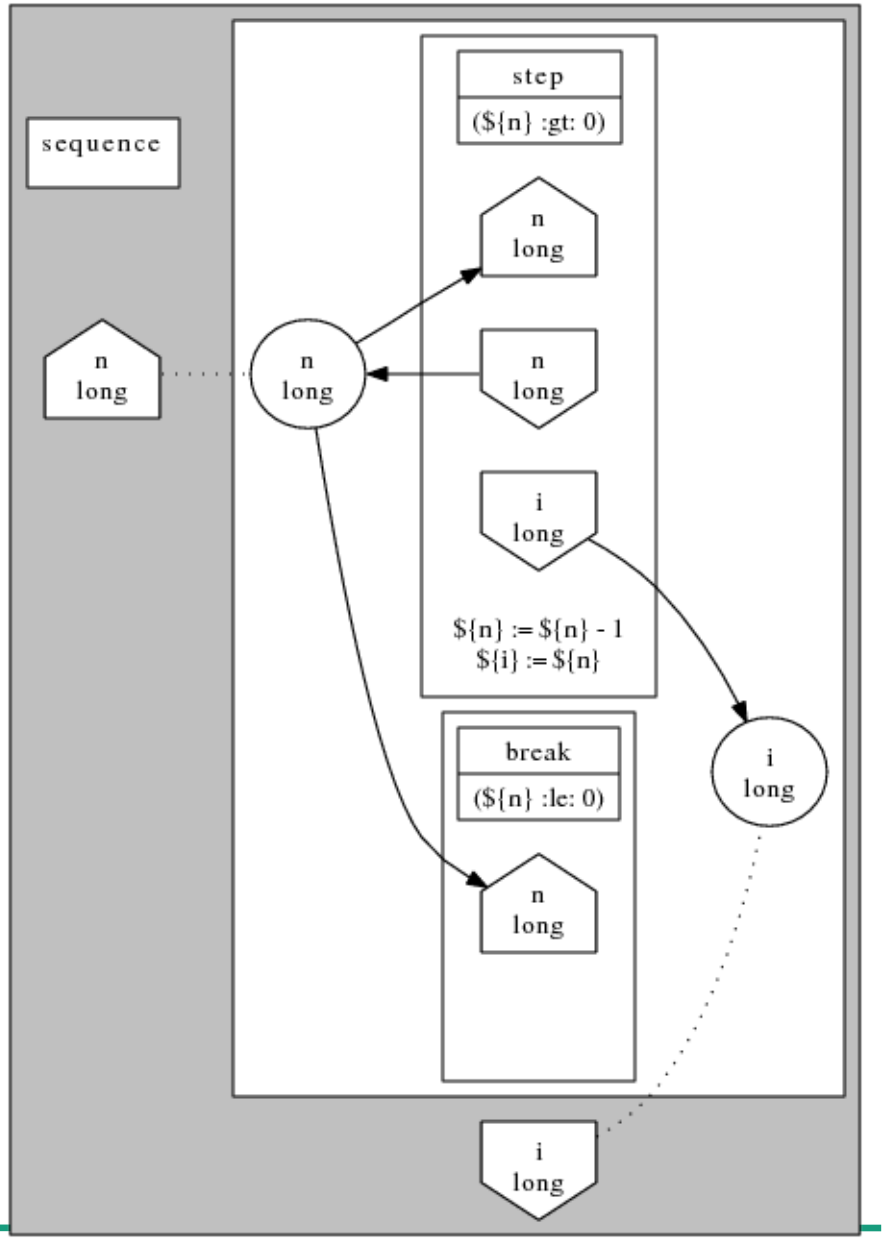
GPI-Space – sum with expression

```
<defun name="sum_expr">

  <include-structs href="pair.xpnet"/>

  <in name="p" type="pair"/>
  <out name="s" type="long"/>

  <expression>
    ${s} := ${p.x} + ${p.y}
  </expression>

</defun>
```



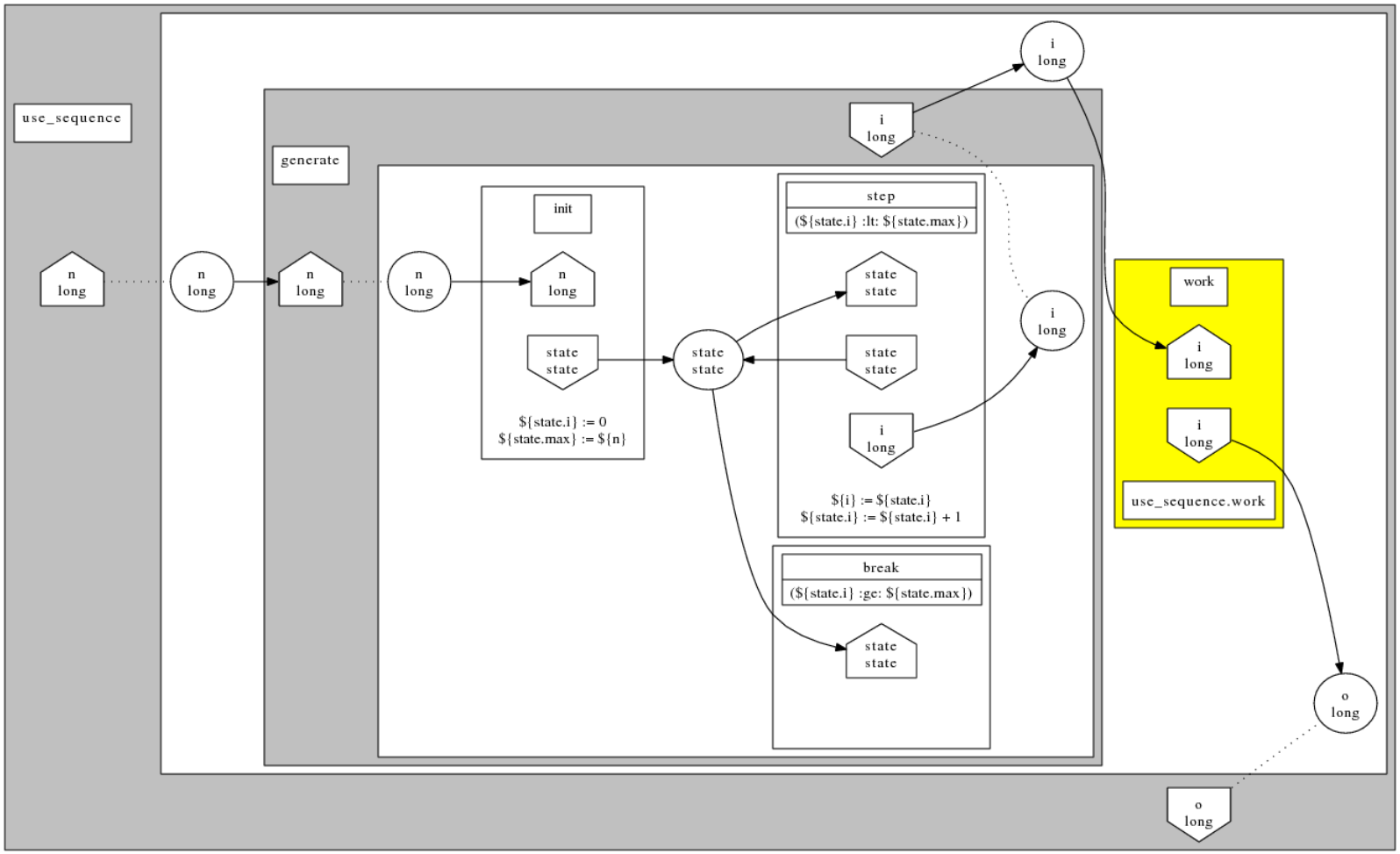
- expressions, user defined types

GPI-Space – sequence



- conditions, inout

GPI-Space – use sequence



- (no)inline

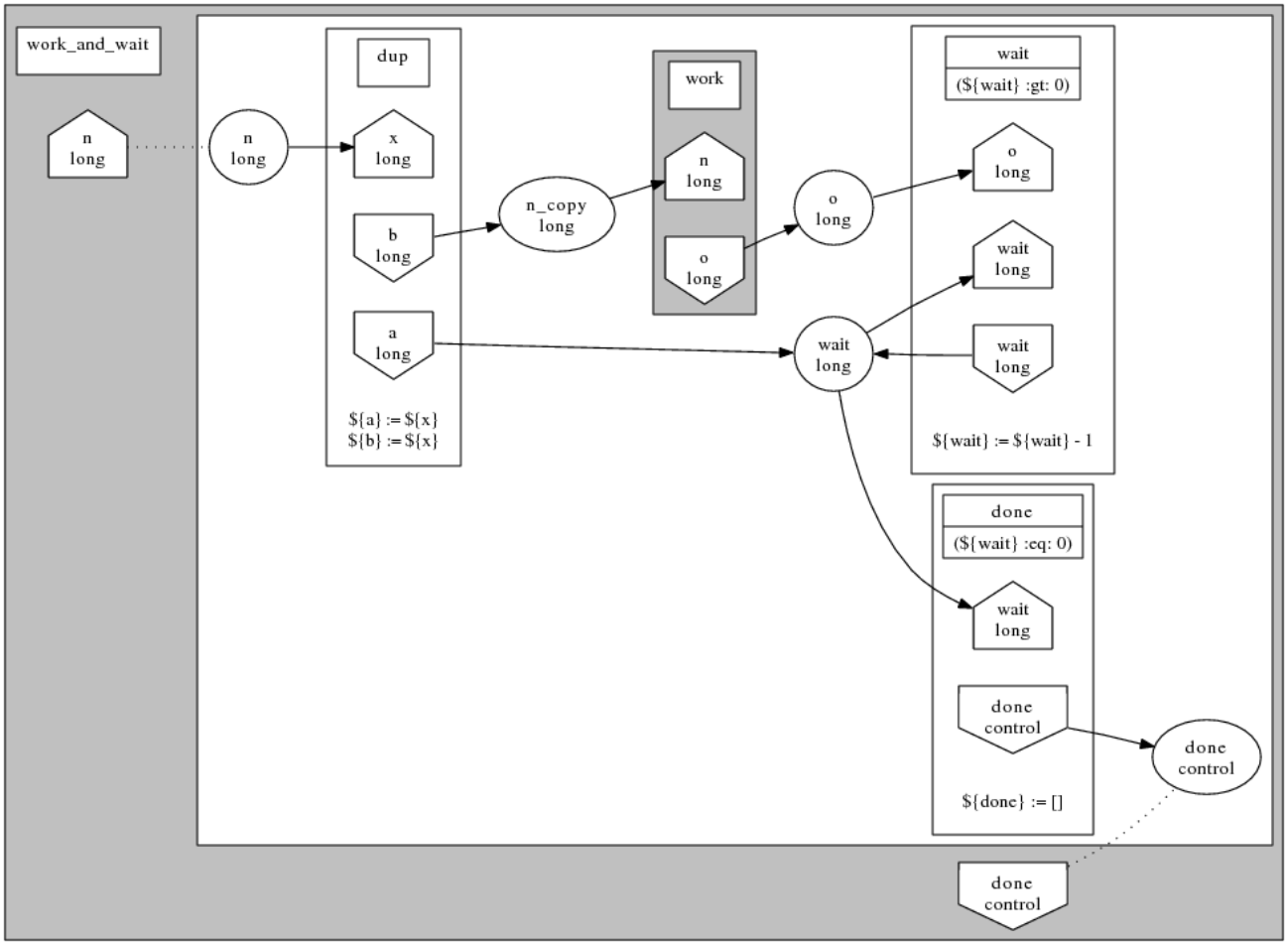
GPI-Space – dup something

```
<template name="dup">

  <in name="x" type="T"/>
  <out name="a" type="T"/>
  <out name="b" type="T"/>

  <expression>
    ${a} := ${x}; ${b} := ${x}
  </expression>

</template>
```

```
<include-template href="dup.xpnet"/>
<specialize name="dup_long" use="dup">
  <type-map replace="T" with="long"/>
</specialize>

<transition name="dup">
  <use name="dup_long"/>
  <connect-in port="x" place="n"/>
  <connect-out port="a" place="wait"/>
  <connect-out port="b" place="n_copy"/>
</transition>
```

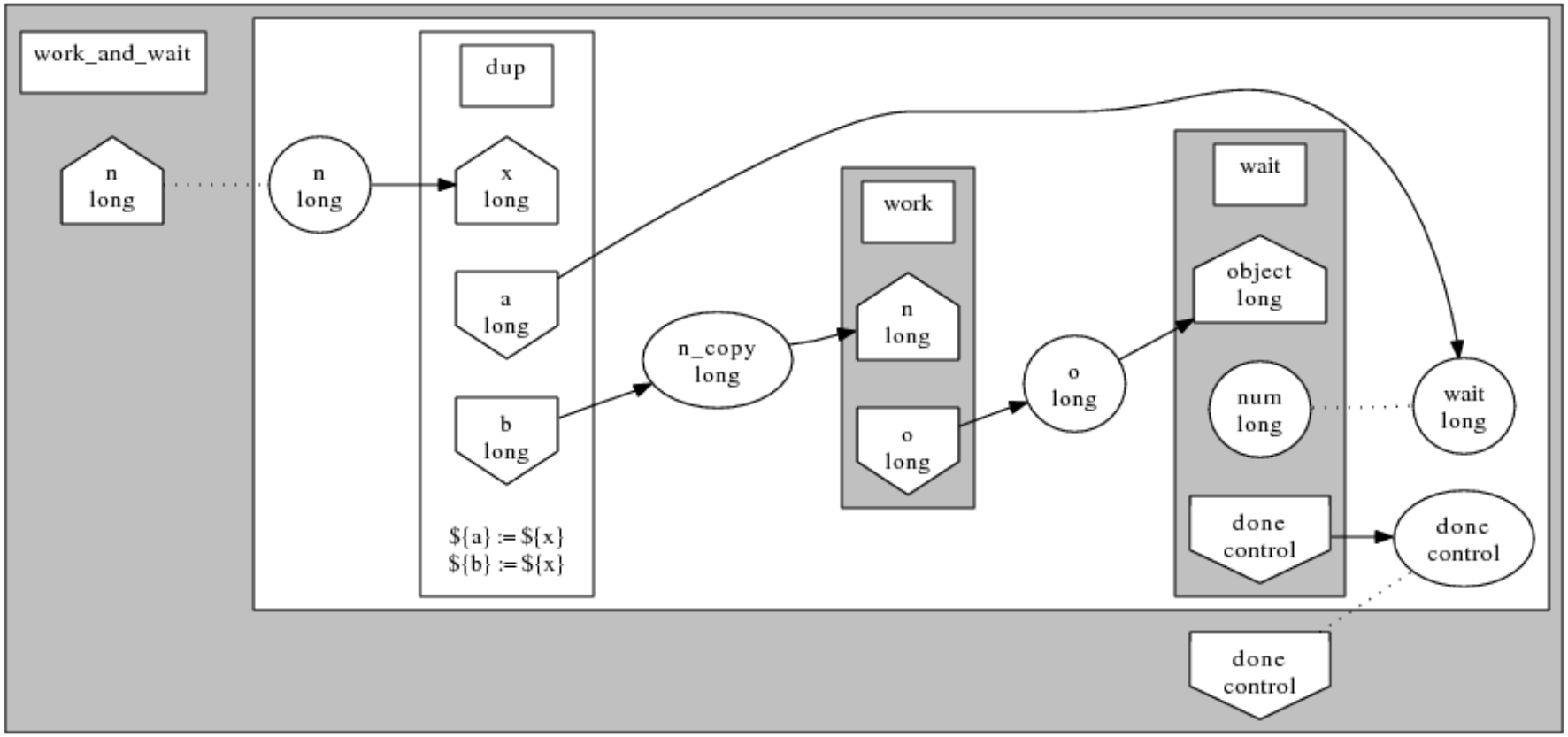
- template

GPI-Space – work and wait



- use template

GPI-Space – work and wait, wait as subnet



- virtual places