Index

abstract factory pattern, see design patterns
abstraction view, 30
ACE, 216
Adapt, 213
adaptive, 211
ADEPT, 250
algebra, 19
algebraic compactness, 23
algorithmic skeleton, 87
anacleto (P3L compiler), 161, 165–168
anamorphism, 20
AOP, see Aspect-oriented programming
application programmer, 61
application-specific integrated circuits (ASICs), 275–276
applications
computer vision, 122
in Eden, 107–116
conjugate gradient, 111–112
Euler numbers, 108–109
Karatsuba algorithm, 109–111
matrix multiplication, 113–115
pair interactions, 115–116
ray tracing, 108
solving PDEs, 175, 179, 183
ASIC, see Application-specific integrated circuits
Aspect Oriented Programming, 211, 214
aspect weaving, 194, 203
AspectIX, 214
AspectJ, 203, 209, 214
associativity, 17, 63, 76
attributes, 284–286
B2B technologies, 248–252
base-object, 199, 205
BBN, 211
BEA Collaborate, 252
behavioural, see levels of abstraction
Beowulf, 107
bifunctor, 18
Bird-Meertens formalism, 30, 47
BizTalk, 251
block recursion, 65
branch-and-bound, 60, 172
brittle (in systems), 196
BSP
Library, 187
model, 183
builder pattern, see design patterns
catamorphism, 19
category, 17
CMI, 250, 265
co-design, 275, 277
CoABS, 223
coalgebra, 20
collective operation, 72
collective operations, 29, 31
collective operations performance, 54
combinator, 3, 62
combine operation, 36
combined rule, 49
commercial-off-the-shelf, 193, 195, 197, 198, 200, 207, 211, 213, 215
commonality, 278–280, 282–283
communication-closed layers, 70
commutativity, 76
compilers
Eden, 92
EKTRAN, 129
GHC, 92
Glasgow Haskell Compiler, 179
PMLS, 130
PUFF, 128
Skeleton-based, 126
SkelML, 128
Component Configurator pattern, 212
components, 249
composition, 4, 17
composition pattern, see design patterns
composition transformation rules, 47
compositional programming, 61
Computational Grid, 221
computer vision, 122
concatenation, 4
concurrency, 276, 279–283, 293, 298
concurrent threads
in Eden, 89
configuration item, 207
configuration object, 205
conjugate gradient
in Eden, 111–112
cons, 13
conslist, 13
constructor, 20
ccontext, 202
contract, 205, 207, 209, 210
control-closed blocks, 70
coproduct, 9
CORBA, 199, 201, 211, 221, 249
event channel, 200, 201
Event Service, 197, 202, 204, 208–210, 214
Notification Service, 214
Object Request Broker, 198
Real-time, 197, 213, 215
cost
acquisition, 195
evolution, 195
excessive, 216
maintenance, 195, 196, 215
non-recurring, 193, 196
procurement, 196
cost models, 88
for Eden skeletons, 93–96, 98, 100, 102, 103, 105–107
parameters, 93, 114
cost-optimality, 39
cotuple, 10
cotupling, 9
CPM, 250
CrossFlow, 250
CS method, 13
cXML, 251

DAML, 224
DARPA, 213, 214, 216
data abstraction, 87
data parallel skeletons, 94–100
DCOM, 249
decomposition, 41
decomposition rules, 48
design patterns, 276–279, 282, 289, 291–302
abstract factory, 293–299
builder, 293–299
Component Configurator, 212
composition, 293
documentation, 291
FSM, 291, 293–299
meta-template, 299–302
singleton, 292–293
design reuse, see reuse
design space exploration, 279, 283
destructor, 20
DH architecture-independent implementation, 43, 45
DH distributed implementation, 43
DH implementation, 38
DH implementation on fixed number of processors, 39
DH skeleton, 38
diagonal, 10
differentiated services, 214
DiffServ, see Differentiated Services
distributed real-time and embedded systems, 193
distributed version of function, 43
distribution function, 37
divide-and-conquer, 23, 60, 100–102
dynamic channels
in Eden, 91, 103
Dynamic Connector, 205, 207, 209
dynamic process creation
in Eden, 89
dynamicTAO, 199, 214
EAI, 249
ECA rules, 264
eCO, 251
Eden, 87–118
case studies, 107–116
conjugate gradient, 111–112
Euler numbers, 108–109
Karatsuba algorithm, 109–111
matrix multiplication, 113–115
pair interactions, 115–116
ray tracing, 108
communication channels, 89
compiler, 92
cost models, 93–94
divide-and-conquer, 102
farm, 95, 102
iterate until, 105
map, 94–96, 98
map and reduce, 100
pipe, 103
replicated workers, 98, 102
ring, 107
self-service farm, 96
torus, 106
data parallel skeletons, 94–100
dynamic channels, 91
dynamic process creation, 89
implementation, 92
iteration protocol, 89
many-to-one communication, 91
merge, 91
non-determinism, 91
noPe, 92
process abstraction, 89, 90
process allocation, 92
process instantiation, 89, 90
run-time system, 92
skeletons, 92–107
dc.par, 101
dc.rw, 101
dc (divide-and-conquer), 100
iterUntil, 104
map.farm, 95
map.par, 89
map.reduce.ssi, 100
map.rw, 98
mr.pm, 99
mr.SSI, 99
mr (map and reduce), 99
pipeD (dynamic channels), 103
pipe.naive, 102
pipe, 102
ring, 106–107
rw (replicated workers), 97
ssf (self-service farm), 96
torus, 105–106
systolic skeletons, 103–107
task parallel skeletons, 100–103
EDIFACT, 250
EFLOW, 250, 265
EJB, 249
elaboration, 287, 289–291
Electronic Data Interchange (EDI), 250
ERP, 249
Euler numbers
in Eden, 108–109
exploration, see design space exploration
factorisation of homomorphisms, 36
factorisation theorem, 7
factory pattern, see design patterns
FAN, 170
farm implementation
of divide-and-conquer, 101
of map, 95–96
of map and reduce, 99
self-service, 96
farmer, 69
fault reconfiguration, 204
FFT, 67
FFT case study, 45
filter, 64
finite state machine pattern, see design patterns
fixed shared data structures, 98
foldr/build rule, 68
FSM pattern, see design patterns
functional, 3
functional abstraction, 87
functional composition, 31
functional programming, 61, 119, 175
functional prototypes, 119
functor, 17
fusion, 68
gather, 72
generative programming, 214
global reduction, 40
Globus, 223, 245
OGSA, 245
H instance, 36, 44
H skeleton, 36
hardware description languages (HDLs), 275–291
SystemC, 279–289, 297–298
Verilog, 280
VHDL, 278–280, 282–289, 291
Haskell, 60, 88, 90, 175
HDC, 60
HDL, see hardware description languages
hierarchy
   inheritance, see paradigms
   module, 280
higher-order function, 2, 60
HOFS
   fold, 121
   map, 121
   Synthesis, 133
homomorphism, 19, 20, 36
hylomorphism, 23
IBM
   MQSeries, 249
   San Francisco, 249
   WebSphere, 252
identity, 4, 17
inheritance, see paradigms
initial object, 19
initiality, 17, 19
injection, 9, 10
integrated services, 215
intellectual property (IP), 275–276, see also reuse
Interceptor pattern, 212
interface
   configuration and control, 200
   functional, 197, 202
   open standard, 197, 198
   quality, 197
   standardised, 202
InterWorkflow, 250
IntServ, see Integrated Services
IP, see intellectual property
iterate until, 103–105
Jacobi iterative method, 175, 179, 186
Java, 214
   EJB, 249
   Real-time, 197, 211
Jini, 244
Karatsuba algorithm
   in Eden, 109–111
Karatsuba’s polynomial multiplication, 67
lazy evaluation, 90
Legion, 223
levels of abstraction
   behavioural, 279–280, 284, 285, 287, 289
   RTL, 276, 279–281, 284
   system, 275–276, 279, 281
libraries, 278–279, 281–286, see also reuse
list catamorphism, 7, 36
list homomorphism, 36
load, 203, 206
localisation schema, 40
many-to-one communication, 91
map, 5, 63, 89, 94–99
map and reduce, 99
map functional, 30
matrix algorithm, 67
matrix multiplication, 134
in Eden, 113–115
maximum segment sum, 10
meta-object, 199, 205, 209, 210
meta-programming, 199–200
meta-template pattern, see design patterns
Michaelson, 119
middleware
   adaptive, 194
   commercial-off-the-shelf, 194
   definition, 193
   dependencies on, 194–199
   primary dependency on, 195, 197
   reflective, 194
   secondary dependency on, 197
   definition, 197
MINERS, 210
mode, 194, 203–204, 207, 210, 212
   transition, 203
module hierarchy, see hierarchy
morphism, 17
MPI, 59, 72
MPI program transformation, 48, 49
MPI program transformations, 47
mss-problem, 10, 44
multi-paradigms, see paradigms
multigrid algorithms, 183
negotiation, 205, 209, 210, 212, 213
non-determinism, 91
non-strictness, 91
Index

OBI, 251
Object Management Group, 213
Object Request Broker, 197–199, 205
object-oriented programming (OOP), 291, 295, see also paradigms
OcamPI3L, 170
OFFER, 249
OIL, 225
OMG, see Object Management Group
OOP, see object-oriented programming
open source, 278
open-binding, 213
optimisation problem, 72
overloading, see paradigms

p-distributed version of function, 37

P3L
  case studies, 169
  compiler, 165–168
  cost model, 155–157
  execution model, 147–148
  FFT-Hist, 150–154, 158–161
  implementation, 161–168
  informal syntax, 149–150
  profiling, 164, 165
  skeleton tree, 151, 153, 158, 165
  skeletons, 146–147
  vs other task and data parallel models, 145

pair interactions
  in Eden, 115–116
paradigms, 278–279, 282, 291, 301–302
  inheritance, 278, 279, 282–286, 293, 297
  overloading, 279, 283, 288–289, 300–301
  polymorphism, 288, 297
  templates, 279, 283, 289–291, 296, 299–302

partial functor, 18
performance model, 50
performance predictability, 49
performance prediction, 44, 52
performance view, 31
Petri-nets, 264
pipeline, 102–103
polymorphism, see paradigms
POPE, 175–179
powerlist, 38
primary dependency, 195, 197

priority, 194
proactive resource allocation, 211
process, see concurrency
process abstraction
  in Eden, 89, 90
process instantiation
  in Eden, 89, 90
process model
  in Eden, 88
product, 9
product category, 18
Profiling
  Dynamic, 132
program design, 52
program design alternatives, 52
program transformation, 122
programmable platform, 275
projection, 9
promotion theorem, 8, 37
prototyping, 119
Proxy pattern, 213
PVM, 179, 182

QoS Language, 207–209
Quality Connector Pattern, 200–213
Quality Object, 199
Quality Objects, 211, 213
Quality of Service, 193, 195, 199, 202, 203, 206, 207, 211, 213, 215
  language, 211
quicksort, 65
QuO, see Quality Objects
Quorum, 211, 213

ray tracing
  in Eden, 108
reactive resource allocation, 211
reduce, 5, 63
reduction decomposed implementation, 42
reduction decomposition, 41
reduction functional, 31
reduction global, 40
reduction implementation, 40
reduction in parts, 41
reduction segmented, 40
reference counting, 69
reflection pattern, 213
reflective, 211
Reflective CCM, 214
register transfer level (RTL), see levels of abstraction
regular structure, see structure relative speedups, 117 replicated workers implementation of divide-and-conquer, 101 of map, 96–98 replicated workers skeleton, 97 reuse, design, 275–276, 278–279, 283–284, 286, 293, 298, 302 ring skeleton, 106–107 RosettaNet, 251 RTL, see levels of abstraction run-time system (RTS), 88, 92 SAT methodology, 30 Scaife, 119 scan, 64 scan functional, 31 scan implementation, 39, 41 scan implementation complexity, 50 scan segmented, 41 scatter, 72 secondary dependency, 197, 203 definition, 197 segmented reduction, 40 segmented scan, 41 SELF-SERV, 250, 266 self-service farm skeleton, 96 self-service implementation farm, 96 of map and reduce, 99 “send-receive considered harmful”, 29 service provider, 224 Services Adapter, 227 Aggregator, 228 Broker, 226 Cost, 226 Discovery, 228 Execution, 229 Fairness, 227 Functional, 225 Management, 225 Mobile, 229 Optimiser, 228 Performance, 227 Provider, 226 Reputation, 229 Security, 226

structure, regular, 283, 287–291
Sun Open Network Environment (ONE), 252
superlinear speedup, 53
swap skeleton, 38
system, see levels of abstraction
System-On-a-Chip (SOC), 275–276, 302
SystemC, see hardware description languages
systolic skeletons, 103–107

TAO, 199, 205, 216
TAP, see Theatre Air Planner
task granularity, 95
task parallel skeletons, 100–103
taskHPf, 170
TBMCS, see Theatre Battle Management Core Systems
templates, see paradigms
Theatre Air Planner, 195
Theatre Battle Management Core Systems, 195
time-optimality, 39
torus skeleton, 105–106
transformation performance impact, 50
transformation rule, combined, 49
transformational programming, 3
transformations for composition, 47
transformations for decomposition, 48
transformations in SAT, 32
transformations, semantics-preserving, 47
Travelling Salesperson Problem, 72
TSP, 72
tupling, 9

UDDI, 251
UML, see Unified Modelling Language
Unified Modelling Language (UML), 278

Value Added Network (VAN), 250
variation, 276, 278–280, 282–283, 286–291
VC, see virtual components
vector, 67
Verilog, see hardware description languages
VHDL, see hardware description languages
virtual components (VCs), 276

WebBIS, 265
WebMethods, 252
WISE, 250
worker, 69
Workflow Management Coalition (WfMC), 264
workflows
cross-enterprise, 249
specification languages, 264
WSDL, 251

X.12, 250
XML, 207, 208
B2B Frameworks, 251
cXML, 251
WSDL, 251

zip functional, 38