Multi-process systems control the behavior of everything from datacenters storing our information to banking systems managing money. Each one of these processes has a prescribed role, their contract, that governs their behavior during the joint computation. When a single process violates their communication contract, the impact of this misbehavior can rapidly propagate through the system. This thesis develops techniques for dynamically monitoring expressive classes of concurrent contracts. We provide multiple mechanisms to monitor contracts of increasing complexity. In order to model message-passing concurrent computation, we use a session type system. First, we present a method for dynamic monitoring and blame assignment where communication contracts are expressed using session types. Second, we describe contract-checking processes that handle stateful contracts that cannot be expressed with a session type. These contract-checking processes are also able to encode type refinements. Third, we encode dependent types in our system which allow us to monitor complex invariants. Finally, we survey a number of other monitoring extensions including a mechanism to monitor deadlock.